app.R

dpandey5

Wed Jun 14 22:16:34 2017

library(shiny)  
library(shinythemes)  
  
library(shinydashboard)

##   
## Attaching package: 'shinydashboard'

## The following object is masked from 'package:graphics':  
##   
## box

library(DT)

##   
## Attaching package: 'DT'

## The following objects are masked from 'package:shiny':  
##   
## dataTableOutput, renderDataTable

library(data.table)  
  
files = list.files("data/", pattern = "\*.csv")  
hospital\_table = as.data.frame(do.call(rbind, lapply(files, fread)))  
  
  
dashboardHeader <- dashboardHeader(disable = TRUE)  
dashboardSidebar <- dashboardSidebar(disable = TRUE, sidebarMenu(  
 menuItem(  
 "Histogram",  
 tabName = "dashboard",  
 icon = icon("dashboard")  
 ),  
 menuItem("Widgets", tabName = "widgets", icon = icon("th"))  
))  
  
  
  
histogram <- tabItem(tabName = "dashboard",  
 fluidRow(  
 box(  
 plotOutput("distPlot1", height = 250),  
 solidHeader = TRUE,  
 collapsible = TRUE  
 ),  
   
 box(  
 title = "Controls",  
 sliderInput("slider", "Number of observations:", 1, 10, 5),  
 solidHeader = TRUE,  
 collapsible = TRUE  
 )  
 ))  
  
widget <- tabItem(tabName = "widgets", h2("Widgets tab content"))  
  
  
  
tabItems <- tabItems(histogram, widget)  
  
dashboardBody <- dashboardBody(tabItems)  
  
# Define UI for application that draws a histogram  
ui <- shinyUI(  
 navbarPage(  
 "My Application",  
 theme = shinytheme("cerulean"),  
 #shinythemes::themeSelector(),  
   
 fluid = TRUE,  
 tabPanel('Hospital Data', sidebarLayout(  
 sidebarPanel(  
 width = 2,  
 checkboxGroupInput(  
 'show\_vars',  
 'Columns in Hospital to show:',  
 names(hospital\_table),  
 selected = names(hospital\_table)  
 )  
 ),  
 mainPanel(DT::dataTableOutput('ex1'))  
 )),  
 tabPanel("Component 1",  
   
 sidebarLayout(  
 sidebarPanel(  
 width = 4,  
 fileInput(  
 'file1',  
 'Choose CSV File',  
 accept = c('text/csv',  
 'text/comma-separated-values,text/plain',  
 '.csv')  
 ),  
 tags$hr(),  
 checkboxInput('header', 'Header', TRUE),  
 radioButtons('sep', 'Separator',  
 c(  
 Comma = ',',  
 Semicolon = ';',  
 Tab = '\t'  
 ),  
 ','),  
 radioButtons(  
 'quote',  
 'Quote',  
 c(  
 None = '',  
 'Double Quote' = '"',  
 'Single Quote' = "'"  
 ),  
 '"'  
 )  
 ),  
 mainPanel(tableOutput('contents'))  
 )),  
 tabPanel("Component 2", sidebarLayout(  
 sidebarPanel(sliderInput(  
 "bins",  
 "Number of bins:",  
 min = 1,  
 max = 50,  
 value = 30  
 )),  
   
 mainPanel(plotOutput("distPlot"))  
 )),  
 tabPanel(  
 "Component 3",  
 dashboardPage(dashboardHeader, dashboardSidebar, dashboardBody)  
 )  
 )  
)  
  
# Define server logic required to draw a histogram  
server <- shinyServer(function(input, output) {  
 output$ex1 <-  
 DT::renderDataTable(DT::datatable(hospital\_table[, input$show\_vars, drop = FALSE],  
 options = list(pageLength = 25)))  
   
   
 output$contents <- renderTable({  
 # input$file1 will be NULL initially. After the user selects  
 # and uploads a file, it will be a data frame with 'name',  
 # 'size', 'type', and 'datapath' columns. The 'datapath'  
 # column will contain the local filenames where the data can  
 # be found.  
   
 inFile <- input$file1  
   
 if (is.null(inFile))  
 return(NULL)  
   
 read.csv(  
 inFile$datapath,  
 header = input$header,  
 sep = input$sep,  
 quote = input$quote  
 )  
 })  
   
 output$distPlot <- renderPlot({  
 # generate bins based on input$bins from ui.R  
 x <- faithful[, 2]  
 bins <- seq(min(x), max(x), length.out = input$bins + 1)  
   
 # draw the histogram with the specified number of bins  
 hist(x,  
 breaks = bins,  
 col = 'darkgray',  
 border = 'white')  
 })  
   
 histdata <- rnorm(500) \* 100  
 histdata[histdata < 0] = histdata[histdata < 0] \* -1  
   
 output$distPlot1 <- renderPlot({  
 # generate bins based on input$bins from ui.R  
 x <- faithful[, 2]  
 bins <-  
 seq(min(histdata), max(histdata), length.out = input$slider + 1)  
   
 # draw the histogram with the specified number of bins  
 hist(histdata, bins,  
 col = 'darkgray',  
 border = 'white')  
 })  
   
})  
  
# Run the application  
shinyApp(ui = ui, server = server)

Shiny applications not supported in static R Markdown documents