***R Shiny***

* Open Source web application framework for R, developed by RStudio.
* Shiny makes it easy to turn analytical analysis into stylish, interactive web apps, presentable to a wider audience.
* Link to few examples of interactive web apps made using shiny
* http://www.showmeshiny.com
* http://shiny.rstudio.com/gallery/

**Needed to build the app using Shiny.**

"shiny" package installed in RStudio.

Yes, you need not be an expert in HTML, CSS or JavaScript, little knowledge would work. However, Shiny makes it simple and easy. We will use RStudio for programming.

**Get Started: ui.r and server.r**

**install.packages("shiny")**

Create user interface in UI.R file,

Control the layout, appearance, widgets that capture user inputs. Also, display the output. E.g., the title, page layout, text input, radio buttons, drop down menus, graphs.

Create the SERVER.R for computation purpose,

Set of instructions that uses the input provided by the user, process them and provides the required output which is further displayed by UI.R script.

**Structure of Shiny Scripts.**

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny) # Load the shiny package  # Define UI for the shiny application here  shinyUI(fluidPage(  titlePanel(), # Application title  sidebarLayout(  sidebarPanel(), # Sidebar panel  mainPanel() # Main Panel  )  )  ) | library(shiny)  shinyServer(  function(input, output) {}  ) |

**Running the Shiny App**

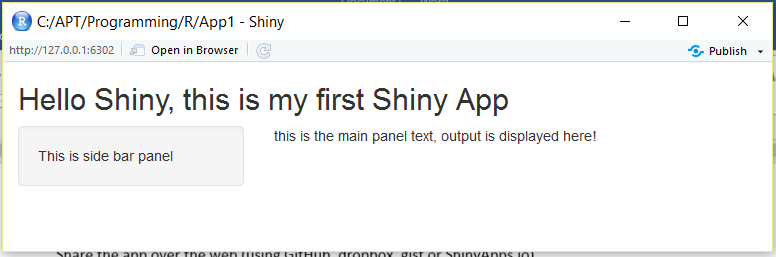
Save both the files UI.R and SERVER.R in R working directory or any other folder. Run the app in the local system using runApp() command specifying the folder name or use the runApp icon from RStudio, there is option to run the app in the browser as well.

Share the app over the web (using GitHub, dropbox, gist or ShinyApps.io)

**Example 1 - Sample Shiny App**

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = 'Hello Shiny, this is my first Shiny App'),  sidebarLayout(  sidebarPanel('This is side bar panel'),  mainPanel('this is the main panel text, output is displayed here!')  )  )) | library(shiny)  shinyServer(  function(input, output) {}  ) |

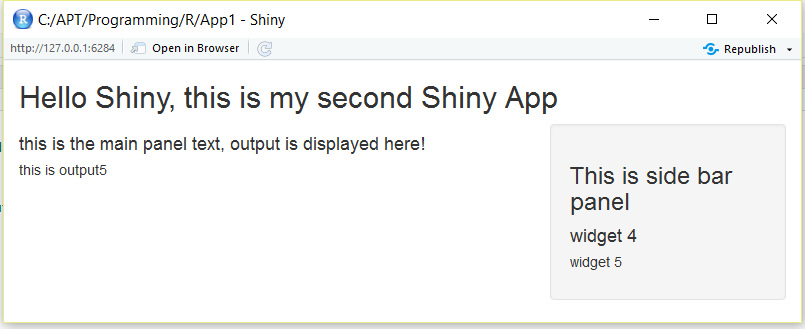
**Output:**



**Example 2 - Sample Shiny App 2**

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = 'Hello Shiny, this is my second Shiny App'),  sidebarLayout(position='right',  sidebarPanel(h3('This is side bar panel'),  h4('widget 4'), h5('widget 5')),  mainPanel(h4('this is the main panel text, output is displayed here!'),  h5('this is output5'))  )  )) | library(shiny)  shinyServer(  function(input, output) {}  ) |

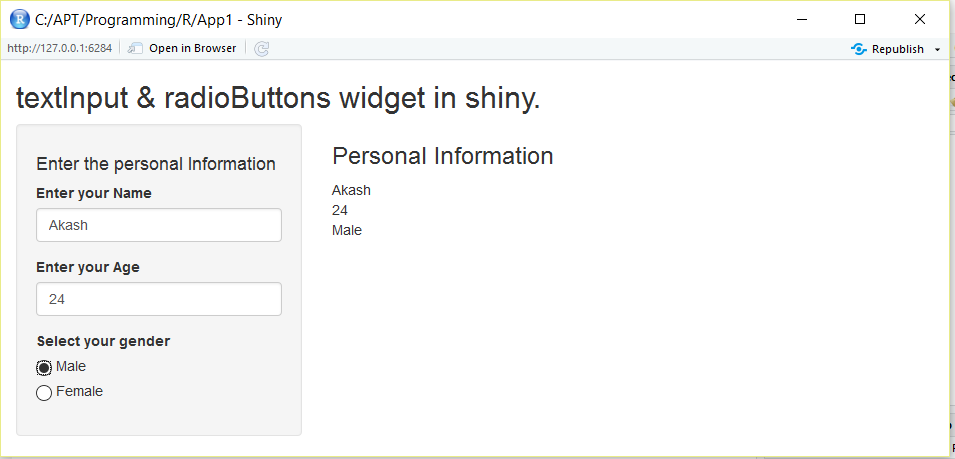
**Output:**



**Example 3 – Text Input & Radio Button widget in Shiny**

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = 'textInput & radioButtons widget in shiny.'),  sidebarLayout(  sidebarPanel(h4('Enter the personal Information'),  textInput('name', 'Enter your Name', ''),  textInput('age', 'Enter your Age', ''),  radioButtons('gender', 'Select your gender', list('Male','Female'), '')),  mainPanel(h3('Personal Information'),  textOutput('myname'),  textOutput('myage'),  textOutput('mygender'))  )  )) | library(shiny)  shinyServer(  function(input, output) {  output$myname = renderText(input$name)  output$myage = renderText(input$age)  output$mygender = renderText(input$gender)  }  ) |

**Output:**



**Example 4 – Slider Input widget in Shiny**

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = 'SliderInput widget in shiny.'),  sidebarLayout(  sidebarPanel(  sliderInput('slide1', 'Select the value from Slider1', min=0, max=5, value=2, animate=TRUE),  sliderInput('slide2', 'Select the value from Slider2', min=0, max=7, value=3, animate=TRUE, step=0.5),  sliderInput('slide3', 'Select the value from Slider3', min=0, max=10, value=c(6,8), animate=TRUE)  ),  mainPanel(  textOutput('myslide1'),  textOutput('myslide2'),  textOutput('myslide3')  )  )  )) | library(shiny)  shinyServer(  function(input, output) {  output$myslide1 = renderText(paste('You selected the value in Slider1 : ',input$slide1))  output$myslide2 = renderText(paste('You selected the value in Slider2 : ',input$slide2))  output$myslide3 = renderText(paste('Slider3 Value: ',input$slide3))  }  ) |

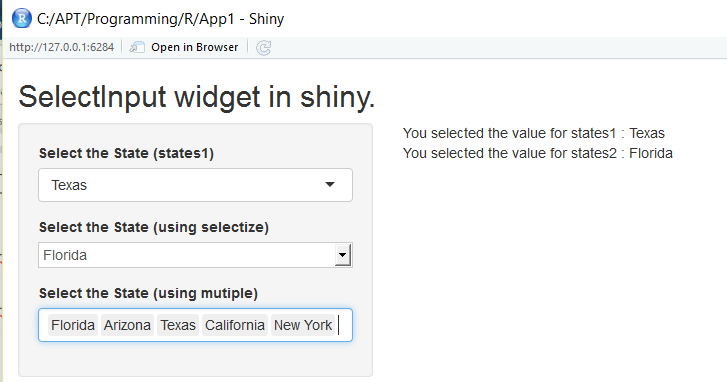
**Output:**



**Example 5 – Select Input widget in Shiny**

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = 'SelectInput widget in shiny.'),  sidebarLayout(  sidebarPanel(  selectInput('states1', 'Select the State (states1)', c('California','Florida','Texas','New York','Arizona'),selected='Texas'),  selectInput('states2', 'Select the State (using selectize)', c('California','Florida','Texas','New York','Arizona'),selectize=FALSE),  selectInput('states3', 'Select the State (using mutiple)', c('California','Florida','Texas','New York','Arizona'),multiple=TRUE)  ),  mainPanel(  textOutput('mystates1'),  textOutput('mystates2')  )  )  )) | library(shiny)  shinyServer(  function(input, output) {  output$mystates1 = renderText(paste('You selected the value for states1 : ',input$states1))  output$mystates2 = renderText(paste('You selected the value for states2 : ',input$states2))  }  ) |

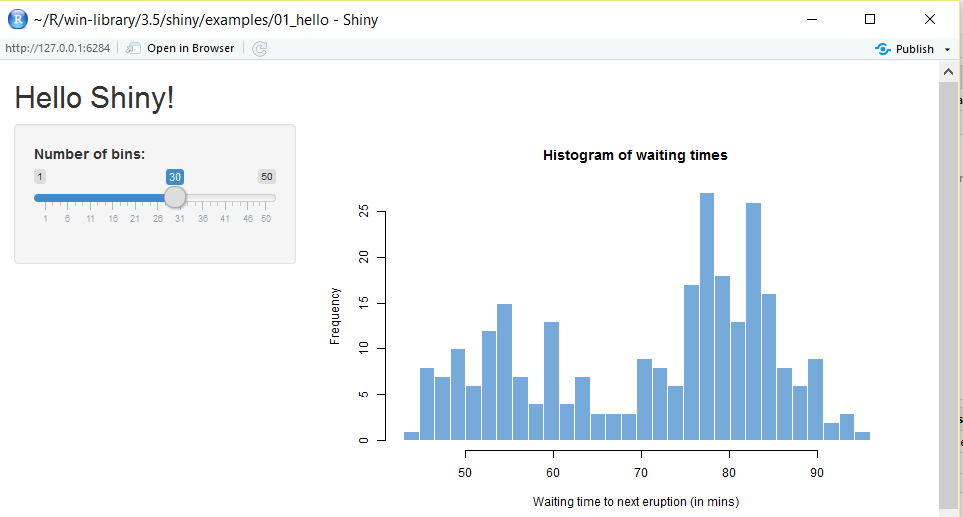
**Output:**



**Example 6 – Shiny Examples**

Try below commands to see in-built examples of shiny.

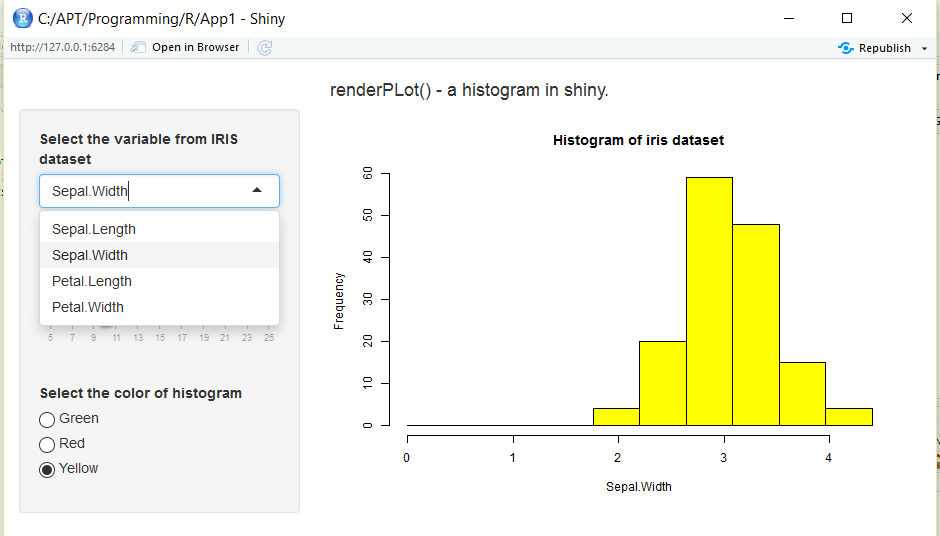
* library(shiny)
* runExample()
  + Valid examples are "01\_hello", "02\_text", "03\_reactivity", "04\_mpg", "05\_sliders", "06\_tabsets", "07\_widgets", "08\_html", "09\_upload", "10\_download", "11\_timer"
* runExample(‘01\_hello’)

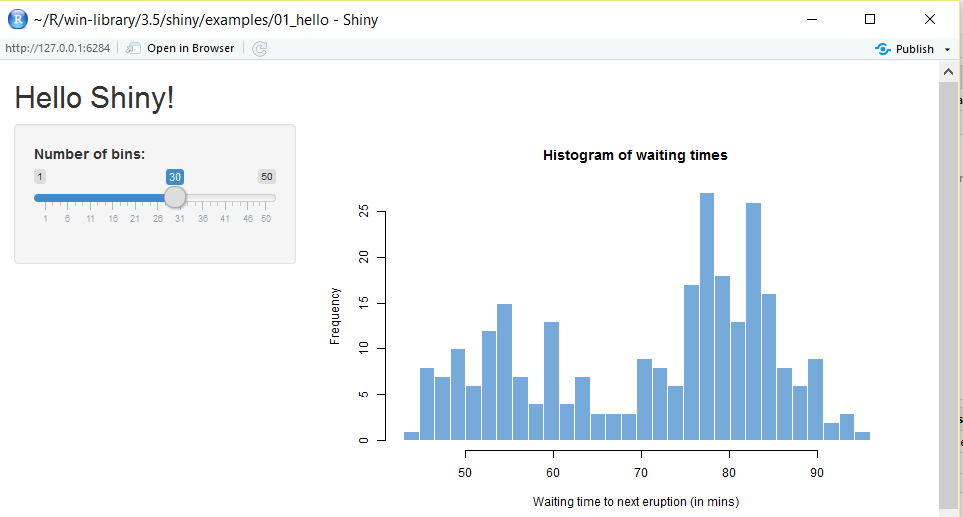


**Example 7 – renderPlot() in Shiny –** to render the plot and plotOutput() to display the plot.

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = h4('renderPLot() - a histogram in shiny.', align='center')),  sidebarLayout(  sidebarPanel(  selectInput('var', 'Select the variable from IRIS dataset', c('Sepal.Length'=1, 'Sepal.Width'=2, 'Petal.Length'=3, 'Petal.Width'=4)),  br(),  sliderInput('bins','Select the number of BINs for histogram', min=5, max=25, value=15),  br(),  radioButtons('colors', 'Select the color of histogram', choices=c('Green','Red','Yellow'), selected='Red')  ),  mainPanel(  plotOutput('myhist')  )  )  )) | library(shiny)  shinyServer(  function(input, output) {  output$myhist = renderPlot({  colm = as.numeric(input$var)  hist(iris[,colm], breaks=seq(0, max(iris[,colm]), l=input$bins+1), col=input$colors, main='Histogram of iris dataset', xlab=names(iris[colm]))  })  }  ) |

**Output:**





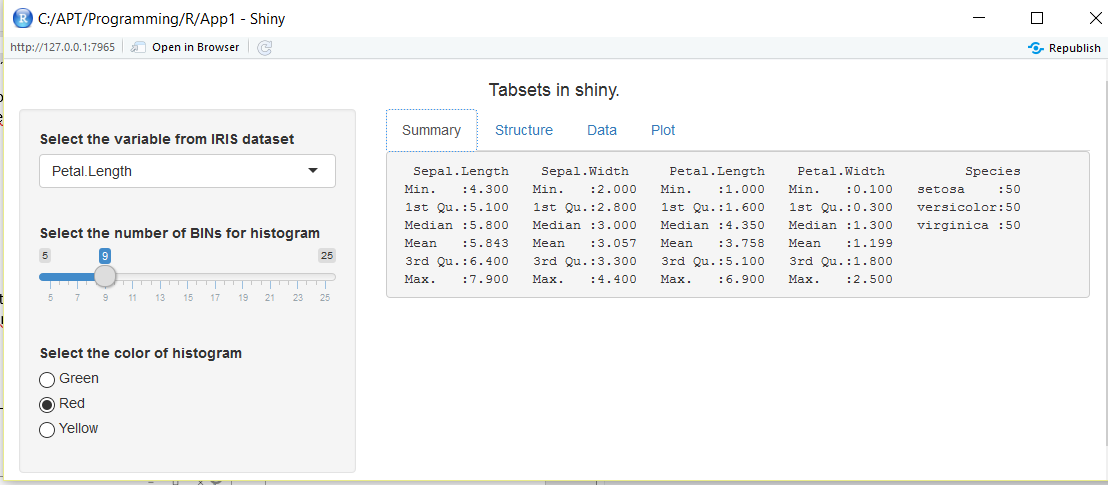
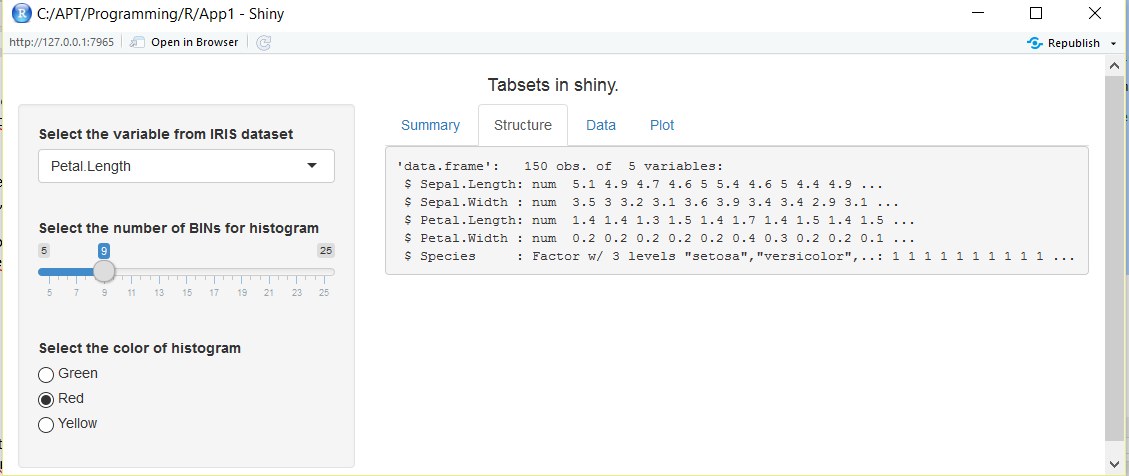
**Example 8 – Tabsets in Shiny**. Other output function such as verbatimTextOutput().

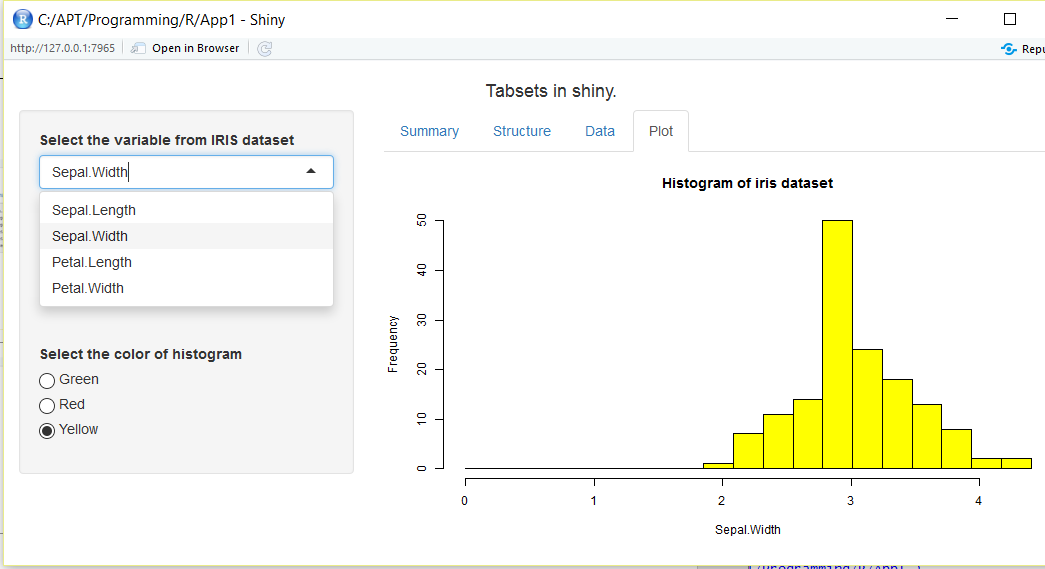
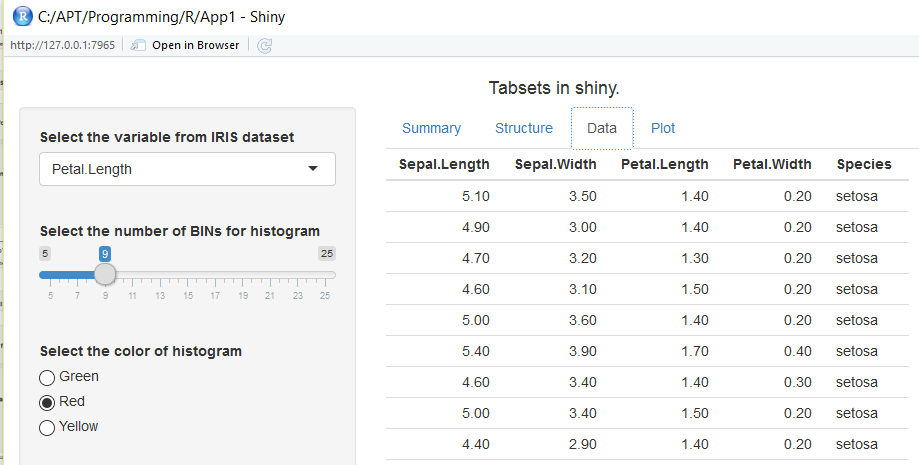
A tab like layout – allowing the user to switch among the different tabs in the main panel.

Usage of renderTable() and tableOutput() to display (output) the information in table format.

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = h4('Tabsets in shiny.', align='center')),  sidebarLayout(  sidebarPanel(  selectInput('var', 'Select the variable from IRIS dataset', c('Sepal.Length'=1, 'Sepal.Width'=2, 'Petal.Length'=3, 'Petal.Width'=4)),  br(),  sliderInput('bins','Select the number of BINs for histogram', min=5, max=25, value=15),  br(),  radioButtons('colors', 'Select the color of histogram', choices=c('Green','Red','Yellow'), selected='Red')  ),  mainPanel(  tabsetPanel(type='tab',  tabPanel('Summary', verbatimTextOutput('summary')),  tabPanel('Structure', verbatimTextOutput('str')),  tabPanel('Data', tableOutput('data')),  tabPanel('Plot', plotOutput('myhist'))  )  )  )  )) | library(shiny)  shinyServer(  function(input, output) {    output$summary = renderPrint({  summary(iris)  })  output$str = renderPrint({  str(iris)  })  output$data = renderTable({  colm = as.numeric(input$var)  #iris[colm]  iris ##to display all columns  })  output$myhist = renderPlot({  colm = as.numeric(input$var)  hist(iris[,colm], breaks=seq(0, max(iris[,colm]), l=input$bins+1), col=input$colors, main='Histogram of iris dataset', xlab=names(iris[colm]))  })  }  ) |

**Output:**

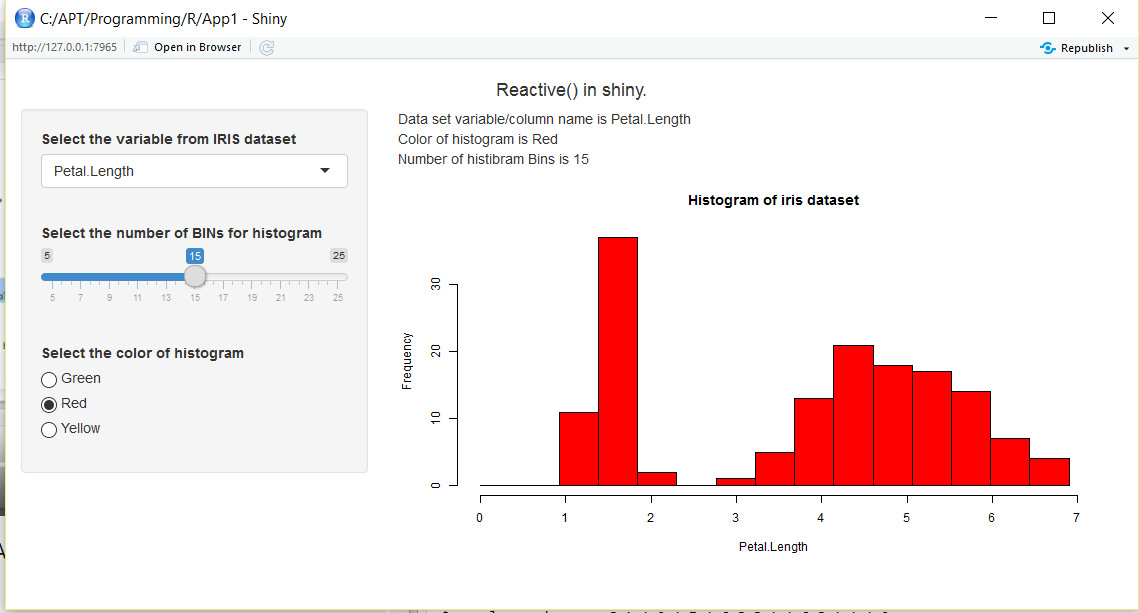




**Example 9 – Reactive() in Shiny**.

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = h4('Reactive() in shiny.', align='center')),  sidebarLayout(  sidebarPanel(  selectInput('var', 'Select the variable from IRIS dataset', c('Sepal.Length'=1, 'Sepal.Width'=2, 'Petal.Length'=3, 'Petal.Width'=4)),  br(),  sliderInput('bins','Select the number of BINs for histogram', min=5, max=25, value=15),  br(),  radioButtons('colors', 'Select the color of histogram', choices=c('Green','Red','Yellow'), selected='Red')  ),  mainPanel(  textOutput('text1'),  textOutput('text2'),  textOutput('text3'),  plotOutput('myhist')  )  )  )) | library(shiny)  data(iris)  shinyServer(  function(input, output) {  colm = reactive({  as.numeric(input$var)  })  output$text1 = renderText({  paste('Data set variable/column name is ', names(iris[colm()]))  })  output$text2 = renderText({  paste('Color of histogram is ', input$colors)  })  output$text3 = renderText({  paste('Number of histibram Bins is ', input$bins)  })  output$myhist = renderPlot({  hist(iris[,colm()], breaks=seq(0, max(iris[,colm()]), l=input$bins+1), col=input$colors, main='Histogram of iris dataset', xlab=names(iris[colm()]))  })  }  ) |

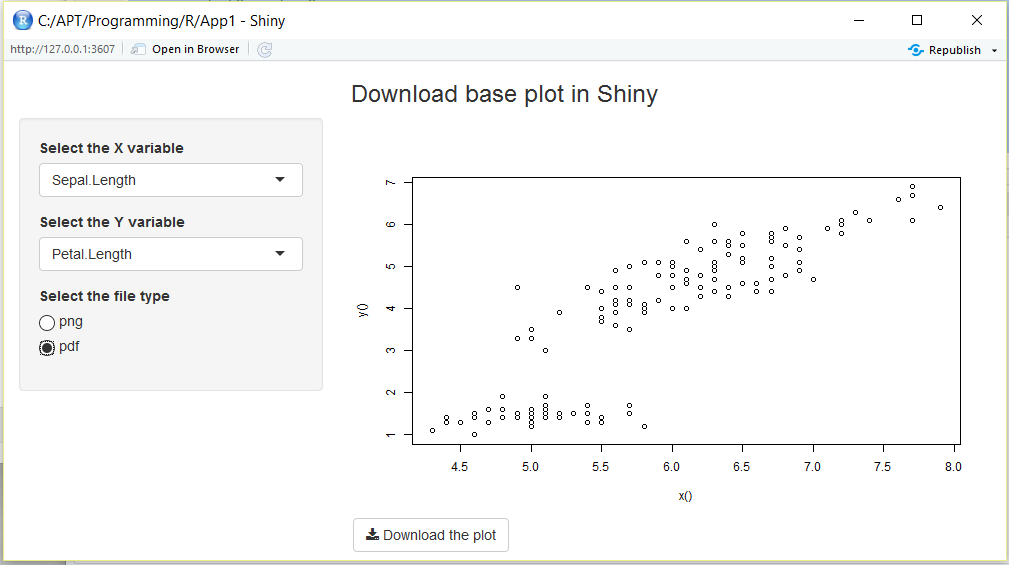
**Output:**



**Example 10 – Download plot in Shiny**.

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = h3('Download base plot in Shiny', align='center')),  sidebarLayout(  sidebarPanel(  selectInput('var1', 'Select the X variable', choices = c('Sepal.Length'=1, 'Sepal.Width'=2, 'Petal.Length'=3, 'Petal.Width'=4)),  selectInput('var2', 'Select the Y variable', choices = c('Sepal.Length'=1, 'Sepal.Width'=2, 'Petal.Length'=3, 'Petal.Width'=4)),  radioButtons('var3', 'Select the file type', choices = list('png','pdf'))  ),  mainPanel(  plotOutput('plot'),  downloadButton(outputId='down', label='Download the plot')  )  )  )) | library(shiny)  data(iris)  shinyServer(  function(input, output) {    x = reactive({  iris[,as.numeric(input$var1)]  })    y = reactive({  iris[,as.numeric(input$var2)]  })    output$plot = renderPlot({  plot(x(),y())  })    output$down = downloadHandler(  # specify the file name  filename = function(){  #iris.png (or) iris.pdf  paste('iris', input$var3, sep='.')  },  content = function(file){  # open device, create the plot, close the device.  # png() or pdf()  if(input$var3 == 'png')  png(file)  else  pdf(file)  plot(x(), y())  dev.off()  }  )    }  ) |

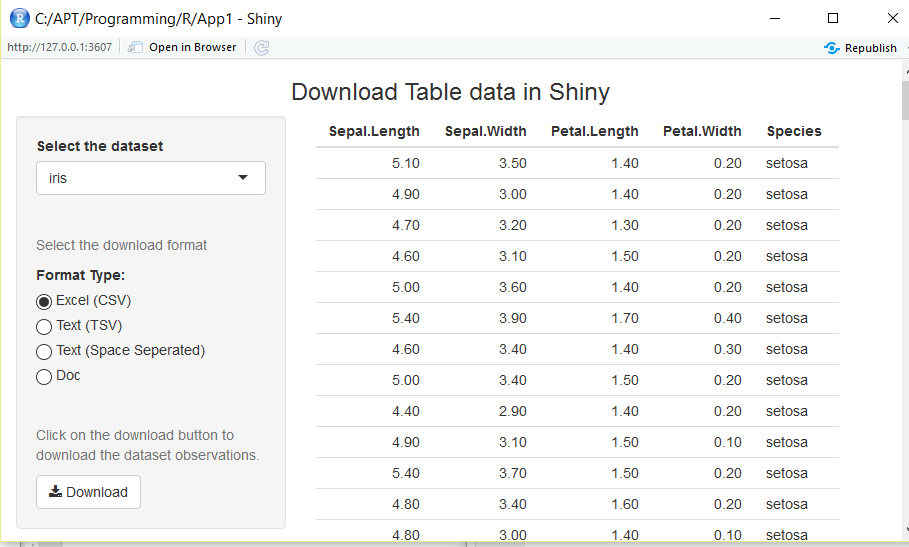
**Output:**



**Example 10 – Download table data in Shiny**.

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = h3('Download Table data in Shiny', align='center')),  sidebarLayout(  sidebarPanel(  selectInput('dataset', 'Select the dataset', choices = c('iris','mtcars','trees')),  br(), helpText('Select the download format'),  radioButtons('type', 'Format Type:', choices = c('Excel (CSV)', 'Text (TSV)', 'Text (Space Seperated)', 'Doc')),  br(), helpText('Click on the download button to download the dataset observations.'),  downloadButton(outputId='downloadData', label='Download')  ),  mainPanel(  tableOutput('table')  )  )  )) | library(shiny)  shinyServer(function(input, output) {    datasetInput = reactive({  switch(input$dataset,  'iris' = iris,  'mtcars'= mtcars,  'trees' = trees)  })    fileext = reactive({  switch(input$type,  'Excel (CSV)' = 'csv', 'Text (TSV)' = 'txt', 'Text (Space Seperated)' = 'txt', 'Doc'= 'doc')  })    output$table = renderTable({  datasetInput()  })    output$downloadData = downloadHandler(  filename = function(){  paste(input$dataset, fileext(), sep='.')  },  content = function(file){  sep = switch(input$type, 'Excel (CSV)' = ',', 'Text (TSV)' = '\t', 'Text (Space Seperated)' = ' ', 'Doc'= ' ')  write.table(datasetInput(), file, sep=sep, row.names = FALSE)  }  )    }  ) |

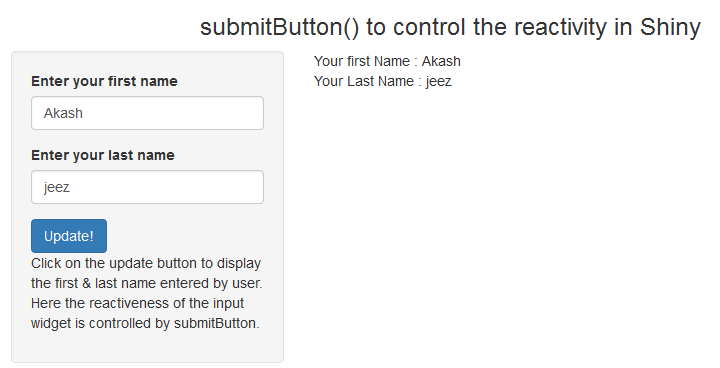
**Output:**



**Example 10 – submitButton() to control the reactivity** **in Shiny**.

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = h3('submitButton() to control the reactivity in Shiny', align='center')),  sidebarLayout(  sidebarPanel(  textInput('text1', 'Enter your first name'),  textInput('text2', 'Enter your last name'),  submitButton(text='Update!'),  p('Click on the update button to display the first & last name entered by user. Here the reactiveness of the input widget is controlled by submitButton.')  ),  mainPanel(  textOutput('txt1'),  textOutput('txt2')  )  )  )) | library(shiny)  shinyServer(function(input, output) {  output$txt1 = renderText(paste('Your first Name : ', input$text1))  output$txt2 = renderText(paste('Your Last Name : ', input$text2))  }  ) |

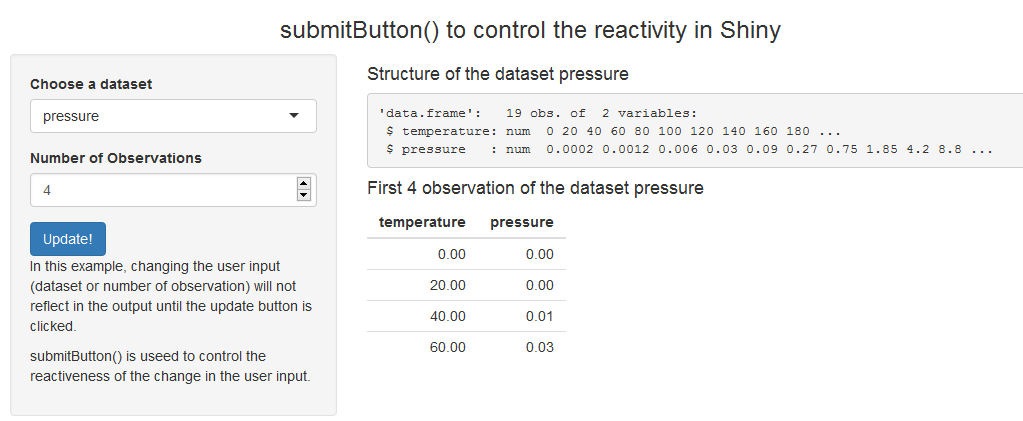
**Output:**



**Example 11 – submitButton() to control the reactivity** **in Shiny**.

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = h3('submitButton() to control the reactivity in Shiny', align='center')),  sidebarLayout(  sidebarPanel(  selectInput('dataset', 'Choose a dataset', choices = c('iris','pressure','mtcars')),  numericInput('obs', 'Number of Observations', 6),  submitButton(text='Update!'),  p('In this example, changing the user input (dataset or number of observation) will not reflect in the output until the update button is clicked.'),  p('submitButton() is useed to control the reactiveness of the change in the user input.')  ),  mainPanel(  h4(textOutput('dataname')),  verbatimTextOutput('structure'),  h4(textOutput('observation')),  tableOutput('view')  )  )  )) | library(shiny)  library(datasets)  shinyServer(function(input, output){    output$dataname = renderText({  paste('Structure of the dataset', input$dataset)  })    output$observation = renderText({  paste('First', input$obs, 'observation of the dataset', input$dataset)  })    output$structure = renderPrint({  str(get(input$dataset))  })    output$view = renderTable({  head(get(input$dataset), n=input$obs)  })  }) |

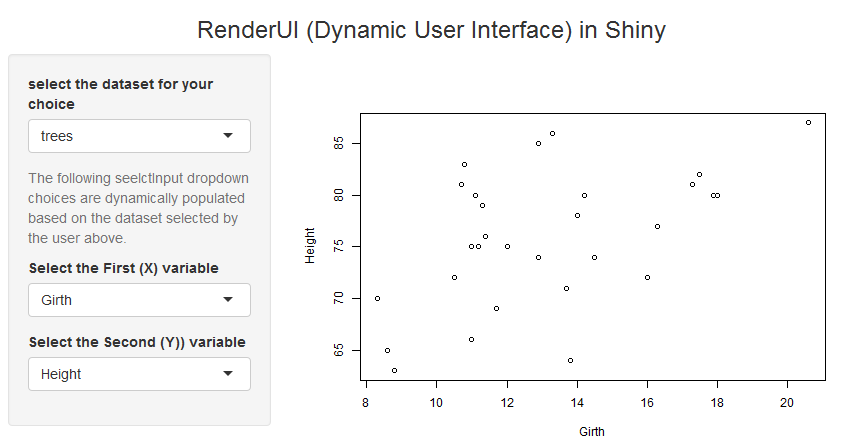
**Output:**



**Example 12 – renderUI() & uiOutput() in Shiny**.

|  |  |
| --- | --- |
| ui.r | Server.r |
| library(shiny)  shinyUI(fluidPage(  titlePanel(title = h3('RenderUI (Dynamic User Interface) in Shiny', align='center')),  sidebarLayout(  sidebarPanel(  selectInput('dataset', 'select the dataset for your choice', choices = c('iris','mtcars','trees')),  helpText('The following seelctInput dropdown choices are dynamically populated based on the dataset selected by the user above.'),  uiOutput('vx'), #vx is coming from renderUI in server.r  uiOutput('vy') #vy is coming from renderUI in server.r  ),  mainPanel(  plotOutput('p')  )  )  )) | library(shiny)  library(datasets)  shinyServer(function(input, output){  var = reactive({  switch(input$dataset,  'iris' = names(iris),  'mtcars' = names(mtcars),  'trees' = names(trees))  })  output$vx = renderUI({  selectInput('variablex', 'Select the First (X) variable', choices = var())  })  output$vy = renderUI({  selectInput('variabley', 'Select the Second (Y)) variable', choices = var())  })  output$p = renderPlot({  attach(get(input$dataset))  plot(x = get(input$variablex), y = get(input$variabley), xlab = input$variablex, ylab = input$variabley)  })  }) |

**Output:**



**ShinyDashboard -** Easy to use Shiny to create dashboards

**## app.R**

**Example 1:**

library(shiny)

library(shinydashboard)

ui = dashboardPage(

dashboardHeader(title='Demo ShinyDashboard package', titleWidth=400),

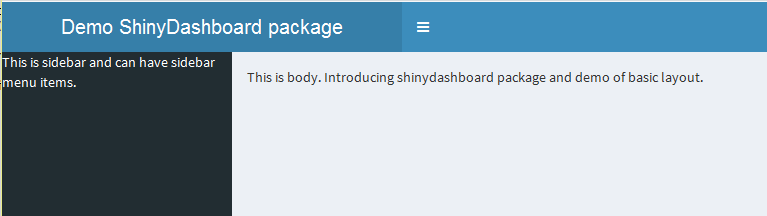
dashboardSidebar('This is sidebar and can have sidebar menu items.', collapsed = TRUE),

dashboardBody('This is body. Introducing shinydashboard package and demo of basic layout.')

)

server = function(input, output){ }

shinyApp(ui, server)



**Example 2:**

library(shiny)

library(shinydashboard)

ui = dashboardPage(

dashboardHeader(title='ShinyDashboard - Create Menu Items', titleWidth=400),

dashboardSidebar(

sidebarMenu(

menuItem(text = 'About', tabName = 'about' , icon = icon('clipboard')),

menuItem(text = 'Date', tabName = 'data', icon = icon('database'), badgeLabel = 'new', badgeColor = 'green'),

menuItem(text = 'Link to Code files', href = 'https://www.google.com', icon = icon('code'))

)

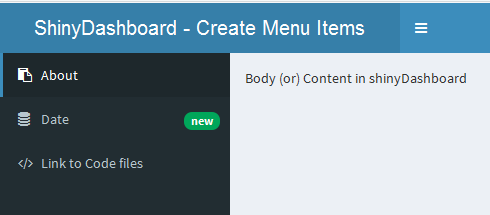
),

dashboardBody('Body (or) Content in shinyDashboard')

)

server = function(input, output){ }

shinyApp(ui, server)



**Example 3 – Add tab pages to menu item**

library(shiny)

library(shinydashboard)

ui = dashboardPage(

dashboardHeader(title='ShinyDashboard - Create Menu Items', titleWidth=400),

dashboardSidebar(

sidebarMenu(

menuItem(text = 'About', tabName = 'about' , icon = icon('clipboard')),

menuItem(text = 'Data', tabName = 'data', icon = icon('database')),

menuItem(text = 'myGithub', href = 'https://github.com/akashjeez/Python-Coding', icon = icon('code'))

)

),

dashboardBody(

tabItems(

tabItem(tabName = 'about', p('This example app demonstrated basic layout of R shnydashboard package'),

p('We will be using mtcars dataset for the sake of demo.')),

tabItem(tabName = 'data', dataTableOutput('mydatatable'))

)

)

)

server = function(input, output, session){

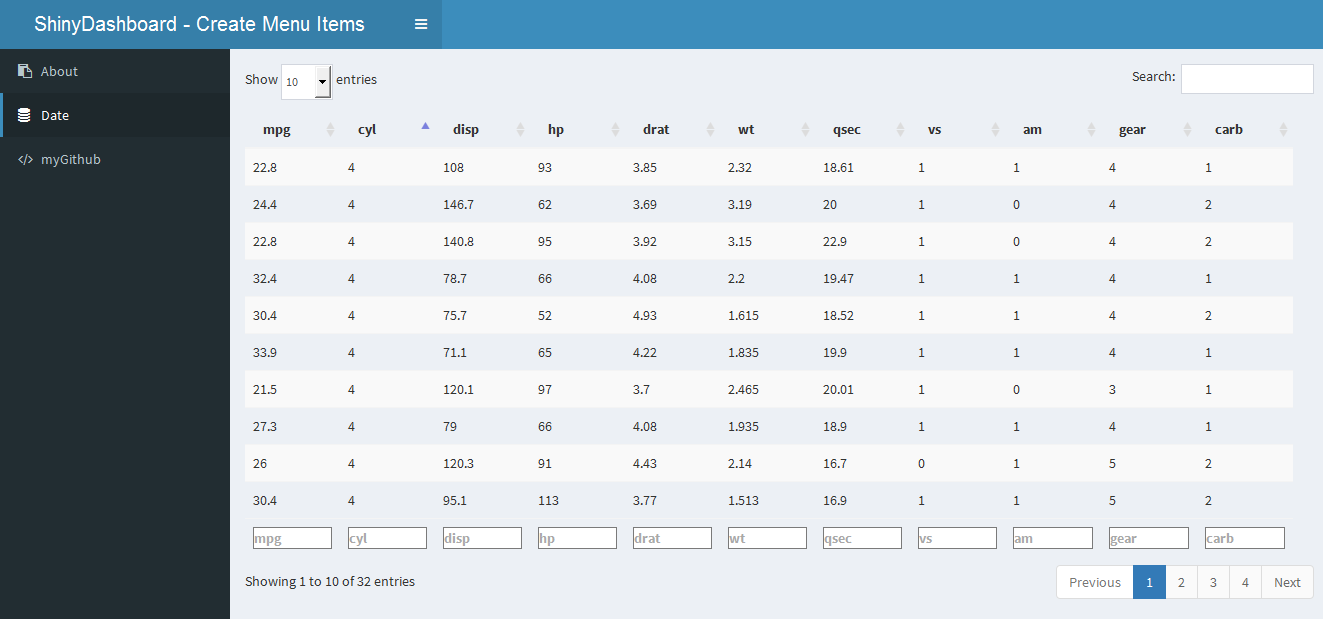
output$mydatatable = renderDataTable({

mtcars

})

}

shinyApp(ui, server)



**Example 4 – Add style to Box**

library(shiny)

library(shinydashboard)

library(plotly)

library(datasets)

ui = dashboardPage(

dashboardHeader(title='ShinyDashboard', titleWidth=230),

dashboardSidebar(

sidebarMenu(

menuItem(text = 'box', tabName = 'box' , icon = icon('check'))

)

),

dashboardBody(

tabItems(

tabItem(tabName = 'box',

fluidRow(box(title = 'Box with a plot', plotlyOutput('plot1', height = 250)),

box(title = 'Box with a plot', plotlyOutput('plot2', height = 250)),

fluidRow(box(title = 'Box with datatable', tableOutput('data'), width = 8),

box(title = 'Box with input widget', uiOutput('inputwidget'), width = 4))

)))

))

server = function(input, output, session){

output$data = renderTable({

head(mtcars)

})

output$plot1 = renderPlotly({

plot\_ly(data = mtcars, x = ~wt, y = ~mpg, type = 'scatter', mode = 'markers')

})

output$plot2 = renderPlotly({

plot\_ly(data = mtcars, x = ~mpg, type = 'histogram')

})

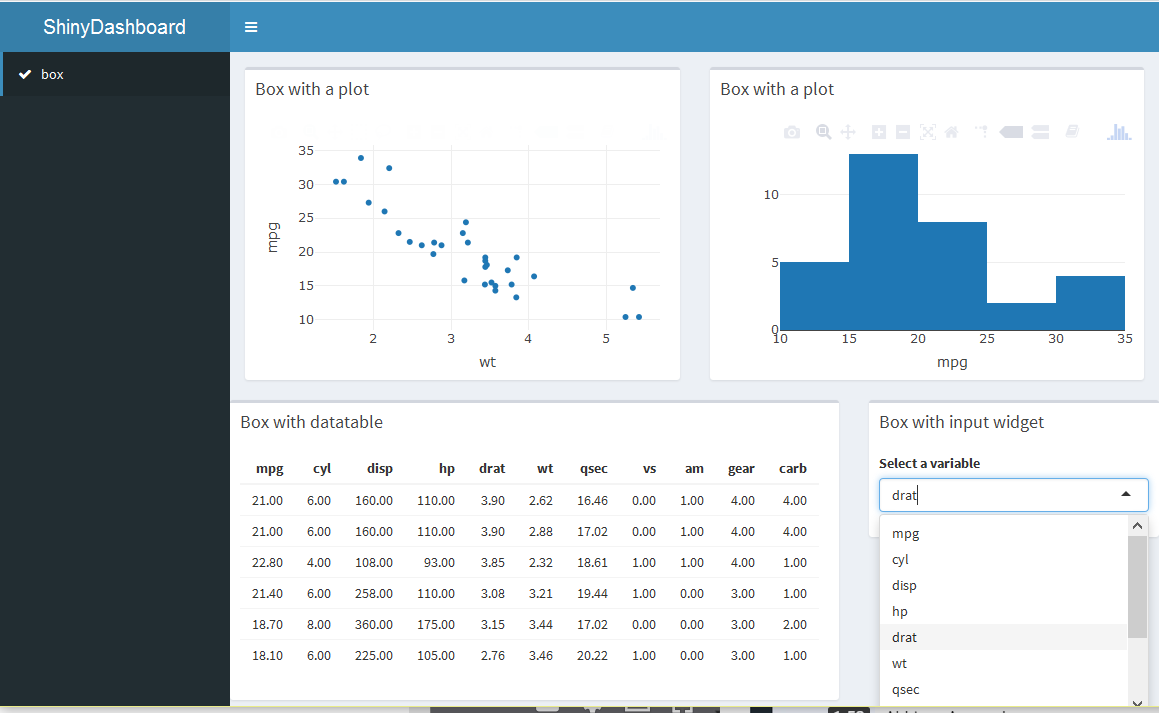
output$inputwidget = renderUI({

selectInput(inputId = 'in', 'Select a variable', choices = names(mtcars))

})

}

shinyApp(ui, server)



**Example 5 – Add Tab Box**

library(shiny)

library(shinydashboard)

library(plotly)

library(datasets)

ui = dashboardPage(

dashboardHeader(title='ShinyDashboard', titleWidth=230),

dashboardSidebar(

sidebarMenu(

menuItem(text = 'tabBox', tabName = 'charts' , icon = icon('check'))

)

),

dashboardBody(

tabItems(

tabItem(tabName = 'charts',

fluidRow(

tabBox(id = 'tabchart1',

tabPanel('Tab1', plotlyOutput('plot1')),

tabPanel('Tab2', plotlyOutput('plot2'))),

tabBox(id = 'tabchart2', side = 'right',

tabPanel('Tab1', 'Tab 1 content'),

tabPanel('Tab2', 'Tab 2 content'))

)

)

)

)

)

server = function(input, output, session){

output$plot1 = renderPlotly({

plot\_ly(data = mtcars, x = ~wt, y = ~mpg, type = 'scatter', mode = 'markers')

})

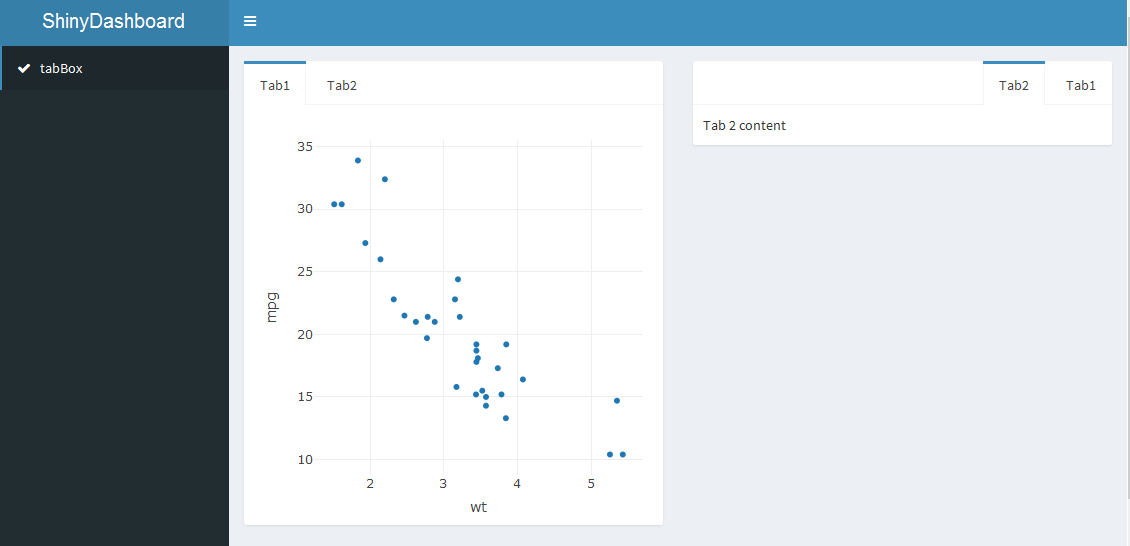
output$plot2 = renderPlotly({

plot\_ly(data = mtcars, x = ~mpg, type = 'histogram')

})

}

shinyApp(ui, server)



**Example 6 – Add Info box**

library(shiny)

library(shinydashboard)

library(plotly)

library(datasets)

ui = dashboardPage(

dashboardHeader(title='ShinyDashboard', titleWidth=230),

dashboardSidebar(

sidebarMenu(

menuItem(text = 'infoBox', tabName = 'IB' , icon = icon('check'))

)

),

dashboardBody(

tabItems(

tabItem(tabName = 'IB',

fluidRow(infoBoxOutput('min\_', width = 3), infoBoxOutput('max\_', width = 3),

infoBoxOutput('sd\_', width = 3), infoBoxOutput('mean\_', width = 3)),

fluidRow(infoBoxOutput('median\_', width = 6)),

fluidRow(infoBoxOutput('inline'), tags$style('#inline {height:75px; line-height:75px;

padding-top:0px; padding-bottom:0px; width:400px;}'))

)

)

)

)

server = function(input, output, session){

s = sample(100:500, 50)

output$min\_ = renderInfoBox({

infoBox(title = 'Minm', value = min(s), subtitle = 'minimum value in dataset', fill = TRUE)

})

output$max\_ = renderInfoBox({

infoBox(title = 'Maxm', value = max(s), subtitle = 'maximum value in dataset', fill = TRUE, color = 'yellow')

})

output$sd\_ = renderInfoBox({

infoBox(title = 'SD', value = round(sd(s),2), subtitle = 'Standard Deviation', icon = icon('arrow-up'))

})

output$median\_ = renderInfoBox({

infoBox(title = tags$b('Median'), value = median(s), subtitle = tags$i('median value in dataset'), icon = icon('angle-double-right'), color = 'purple')

})

output$mean\_ = renderInfoBox({

infoBox(title = 'Mean', value = mean(s), subtitle = 'mean value in dataset', fill = TRUE, icon = icon('angle-double-right'))

})

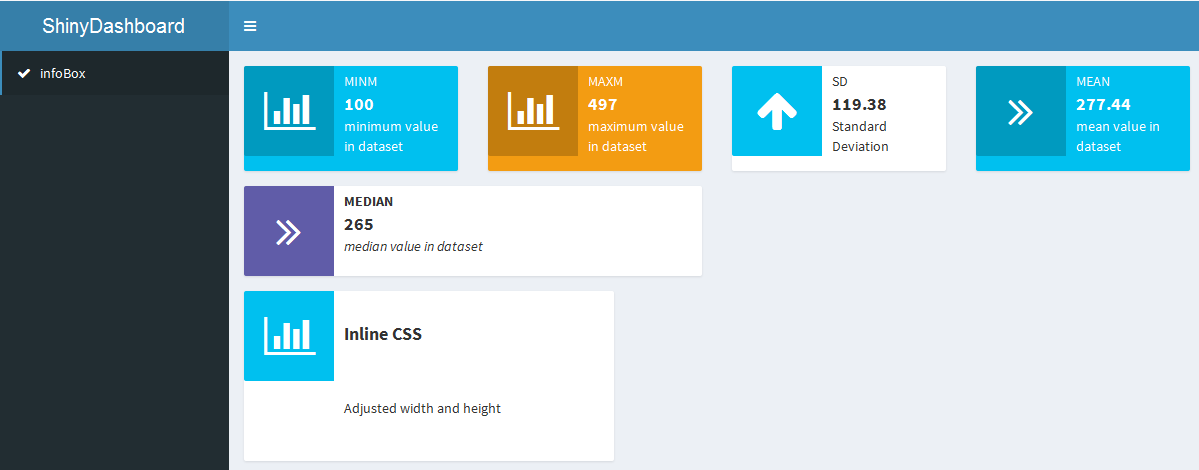
output$inline = renderInfoBox({

infoBox(title = '', value = 'Inline CSS', subtitle = 'Adjusted width and height')

})

}

shinyApp(ui, server)



**Example 7 – Add value Box**

library(shiny)

library(shinydashboard)

library(plotly)

library(datasets)

ui = dashboardPage(

dashboardHeader(title='ShinyDashboard', titleWidth=230),

dashboardSidebar(

sidebarMenu(

menuItem('valuebox', tabName = 'vb')

)

),

dashboardBody(

tabItems(

tabItem(tabName = 'vb',

fluidRow(valueBoxOutput('min', width=3), valueBoxOutput('max', width=3),

valueBoxOutput('sd', width=3)),

fluidRow(valueBoxOutput('median', width=3), valueBoxOutput('mean', width=3))

)

)

)

)

server = function(input, output, session){

s = sample(100:500, 50)

output$min = renderValueBox({

valueBox(value = min(s), subtitle = 'min value in dataset')

})

output$max = renderValueBox({

valueBox(value = max(s), subtitle = 'max value in dataset')

})

output$sd = renderValueBox({

valueBox(value = round(sd(s),2), subtitle = 'standard deviation', icon('arrow-up'))

})

output$mean = renderValueBox({

valueBox(value = mean(s), subtitle = 'mean value in dataset', icon('angle-double-right'))

})

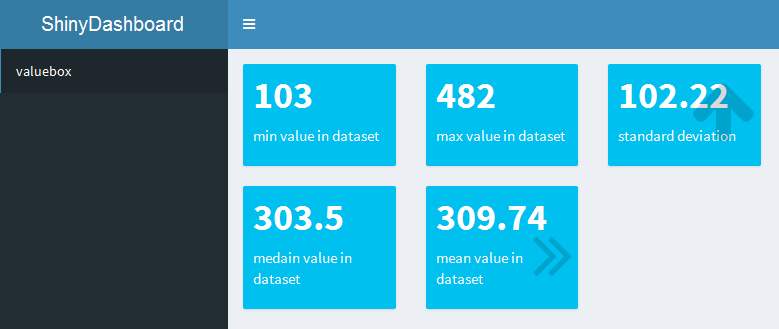
output$median = renderValueBox({

valueBox(value = median(s), subtitle = 'medain value in dataset')

})

}

shinyApp(ui, server)



**Example 8 - Change background color of dashboard header**

library(shiny)

library(shinydashboard)

library(plotly)

library(datasets)

#Available colors are black, green, yellow, red, blue, purple

ui = dashboardPage(skin = 'purple',

dashboardHeader(title='ShinyDashboard', titleWidth=230),

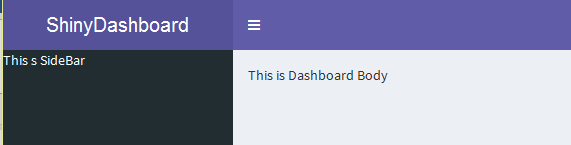
dashboardSidebar('This s SideBar'),

dashboardBody('This is Dashboard Body')

)

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

shinyApp(ui, server)



**Example 9 – Add background color to Info or Box value**

library(shiny)

library(shinydashboard)

library(plotly)

library(datasets)

ui = dashboardPage(skin = 'yellow',

dashboardHeader(title='ShinyDashboard', titleWidth=230),

dashboardSidebar(

sidebarMenu(

menuItem('BoxColor', tabName = 'boxcolor')

)

),

dashboardBody(

tabItems(

tabItem(tabName = 'boxcolor',

fluidRow(

infoBox(title = 'Red', value = '007', color = 'red', width = 3),

infoBox(title = 'Yellow', value = '007', color = 'yellow', width = 3),

infoBox(title = 'Aqua', value = '007', color = 'aqua', width = 3),

infoBox(title = 'Blue', value = '007', color = 'blue', width = 3)

),

fluidRow(

infoBox(title = 'Navy', value = '007', color = 'navy', width = 3, fill = T),

infoBox(title = 'Teal', value = '007', color = 'teal', width = 3, fill = T),

infoBox(title = 'Olive', value = '007', color = 'olive', width = 3, fill = T),

infoBox(title = 'Lime', value = '007', color = 'lime', width = 3, fill = T)

)

)

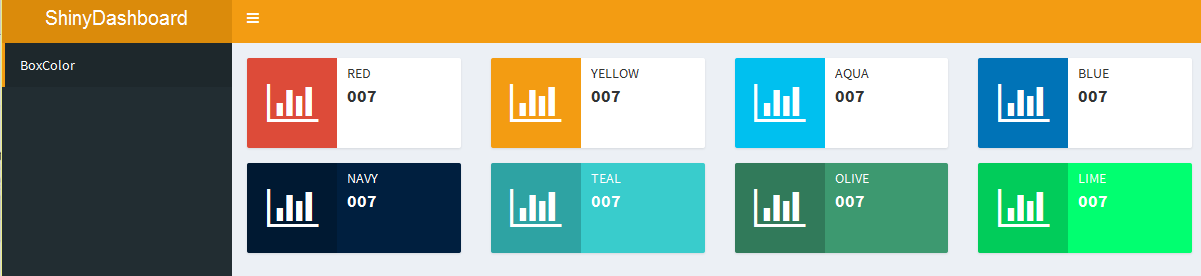
)

)

)

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

shinyApp(ui, server)



**Example 10 – Add icons to sidebar menu items.**

library(shiny)

library(shinydashboard)

library(plotly)

library(datasets)

ui = dashboardPage(skin = 'yellow',

dashboardHeader(title='ShinyDashboard', titleWidth=230),

dashboardSidebar(

sidebarMenu(

menuItem('Dashbaord1', tabName = 'dashboard1', icon('tachometer')),

menuItem('Dashbaord2', tabName = 'dashboard2', icon('tachometer'), class ='fa-lg'),

menuItem('Dashbaord3', tabName = 'dashboard3', icon('tachometer'), class = 'fa-rotate-90'),

menuItem('Dashbaord4', tabName = 'dashboard4', icon('cog'), class = 'fa-spin'),

menuItem('Dashbaord5', tabName = 'dashboard5', icon('cog'), class = 'fa-pulse'),

menuItem('Dashbaord6', tabName = 'dashboard6', icon('cog'), class = 'fa-border'),

menuItem('Dashbaord7', tabName = 'dashboard7', icon('hourglass', lib = 'glyphicon')),

menuItem('Dashbaord8', tabName = 'dashboard8', icon('square')),

menuItem('Dashbaord9', tabName = 'dashboard9', icon('square', class = 'mystyle')),

tags$style('.mystyle {color:yellow;}')

)

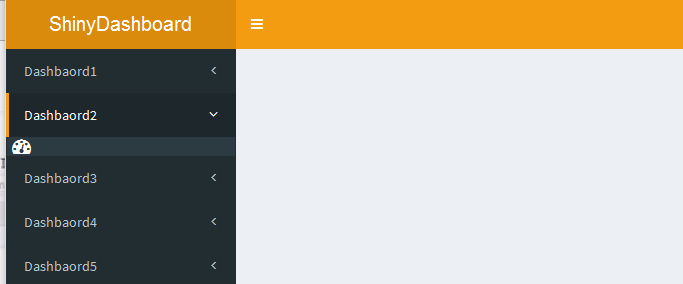
),

dashboardBody()

)

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

shinyApp(ui, server)



**Example 11 – Add Image or Logo in Dashboard Header.**

library(shiny)

library(shinydashboard)

library(plotly)

library(datasets)

#title = tags$a(href = 'https://www.google.com', tags$img(src = 'rstudio.jpg', height = '50', weight = '50'), 'Diamon Explorer')

title = tags$a(href = 'https://www.google.com', icon('diamond'), 'Diamon Explorer')

ui = dashboardPage(skin = 'yellow',

dashboardHeader(title = title, titleWidth = 250),

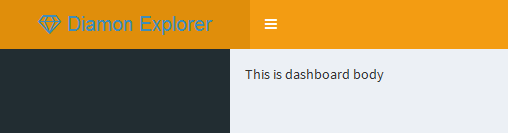
dashboardSidebar(),

dashboardBody('This is dashboard body')

)

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

shinyApp(ui, server)



**Example 12 – Add Social Media icons on right side of dashboard header**

library(shiny)

library(shinydashboard)

library(plotly)

library(datasets)

ui = dashboardPage(skin = 'yellow',

dashboardHeader(title = 'ShinyDashbaord', titleWidth = 230,

tags$li(class='dropdown', tags$a(href='https://www.youtube.com/playlist?list=PLkufgdB5tef59LgDqdqjiNRwnLvlp74XB', icon('youtube'), 'My Channel', target = '\_blank')),

tags$li(class='dropdown', tags$a(href='https://www.linkedin.com/in/akash-ponnurangam-408363125/', icon('linkedin'), 'My Profile', target = '\_blank')),

tags$li(class='dropdown', tags$a(href='https://github.com/akashjeez/Python-Coding', icon('github'), 'Source Code', target = '\_blank'))

),

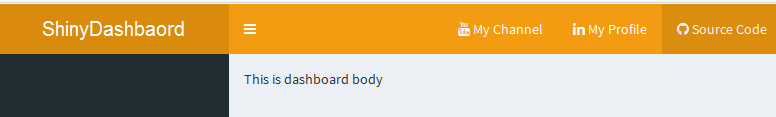
dashboardSidebar(),

dashboardBody('This is dashboard body')

)

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

shinyApp(ui, server)



**Example 13 – Add Popover to Infobox with metrics description.**

library(shiny)

library(shinydashboard)

library(shinyBS) # for popover

library(ggplot2) # for diamonds dataset

ui <- dashboardPage(

dashboardHeader(title = "ShinyDashboard", titleWidth = 400),

dashboardSidebar(

sidebarMenu(

menuItem("Dashboard", tabName = "dashboard", icon = icon("dashboard"))

)

),

dashboardBody(

tags$head(

tags$link(rel = "stylesheet", type = "text/css", href = "custom3.css")

),

bsPopover(id="q1", title = "Mean",

content = "Mean price of diamonds",

trigger = "hover",

placement = "right",

options = list(container = "body")),

bsPopover(id="info2", title = "Median",

content = "Median price of diamonds",

trigger = "hover",

placement = "right",

options = list(container="body")),

tabItems(

tabItem(tabName = "dashboard",

fluidRow(

infoBoxOutput("info1"), # first infoBox

infoBoxOutput("info2") # second infoBox

)

)

)

)

)

server <- function(input, output, session) {

output$info1 <- renderInfoBox({

infoBox("Mean", round(mean(diamonds$price), 2),

icon = icon("usd"),

subtitle = tags$a(icon("question-circle"), id="q1"))

})

output$info2 <- renderInfoBox({

infoBox("Median", median(diamonds$price), icon = icon("usd"))

})

}

shinyApp(ui, server)

