**1.Basic structure of Shiny App:**

library(shiny)

# Define UI ----

ui <- fluidPage(

)

# Define server logic ----

server <- function(input, output) {

}

# Run the app ----

shinyApp(ui = ui, server = server)

**2. Pages:**

(1) fluidPage(): fluid like page

(2) fillPage

(3) fixedPage

(4) navbarPage

navbarPage()

ui <- navbarPage("My Application",

tabPanel("Component 1"),

tabPanel("Component 2"),

tabPanel("Component 3")

)

**3. Layout:**

(1) sidebarLayout: sidebarPanel, mainPanel

sidebarLayout(position = "right"): control the position of sidebar

sidebarPanel(

# Inputs excluded for brevity

),

mainPanel(

# Outputs excluded for brevity

)

)

(2) fluidRow: grid layout, 12 column wide,fluidRow() + column()

fluidRow(

column(4,

wellPanel(

sliderInput("obs", "Number of observations:",

min = 1, max = 1000, value = 500)

)

),

column(8,

plotOutput("distPlot")

)

)

ui <- fluidPage(

title = "Diamonds Explorer",

plotOutput('plot'),

hr(),

fluidRow(

column(3,

h4("Diamonds Explorer"),

sliderInput('sampleSize', 'Sample Size',

min=1, max=nrow(dataset), value=min(1000, nrow(dataset)),

step=500, round=0),

br(),

checkboxInput('jitter', 'Jitter'),

checkboxInput('smooth', 'Smooth')

),

column(4, offset = 1,

selectInput('x', 'X', names(dataset)),

selectInput('y', 'Y', names(dataset), names(dataset)[[2]]),

selectInput('color', 'Color', c('None', names(dataset)))

),

column(4,

selectInput('facet\_row', 'Facet Row', c(None='.', names(dataset))),

selectInput('facet\_col', 'Facet Column', c(None='.', names(dataset)))

)

)

)

(3)tabsetPanel()

mainPanel(

tabsetPanel(

tabPanel("Plot", plotOutput("plot")),

tabPanel("Summary", verbatimTextOutput("summary")),

tabPanel("Table", tableOutput("table"))

)

)

(4) navlistPanel()

ui <- fluidPage(

titlePanel("Application Title"),

navlistPanel(

"Header A",

tabPanel("Component 1"),

tabPanel("Component 2"),

"Header B",

tabPanel("Component 3"),

tabPanel("Component 4"),

"-----",

tabPanel("Component 5")

)

)

(5) navbarMenu()

ui <- navbarPage("My Application",

tabPanel("Component 1"),

tabPanel("Component 2"),

navbarMenu("More",

tabPanel("Sub-Component A"),

tabPanel("Sub-Component B"))

)

4. theme:

5. HTML

|  |  |  |
| --- | --- | --- |
| p | <p> | A paragraph of text |
| h1 | <h1> | A first level header |
| h2 | <h2> | A second level header |
| h3 | <h3> | A third level header |
| h4 | <h4> | A fourth level header |
| h5 | <h5> | A fifth level header |
| h6 | <h6> | A sixth level header |
| a | <a> | A hyper link |
| br | <br> | A line break (e.g. a blank line) |
| div | <div> | A division of text with a uniform style |
| span | <span> | An in-line division of text with a uniform style |
| pre | <pre> | Text ‘as is’ in a fixed width font |
| code | <code> | A formatted block of code |
| img | <img> | An image |
| strong | <strong> | Bold text |
| em | <em> | Italicized text |
| HTML |  | Directly passes a character string as HTML code |

6. widgets

|  |  |
| --- | --- |
| actionButton | Action Button |
| checkboxGroupInput | A group of check boxes |
| checkboxInput | A single check box |
| dateInput | A calendar to aid date selection |
| dateRangeInput | A pair of calendars for selecting a date range |
| fileInput | A file upload control wizard |
| helpText | Help text that can be added to an input form |
| numericInput | A field to enter numbers |
| radioButtons | A set of radio buttons |
| selectInput | A box with choices to select from |
| sliderInput | A slider bar |
| submitButton | A submit button |
| textInput | A field to enter text |

7. output

| Output function | Creates |
| --- | --- |
| dataTableOutput | DataTable |
| htmlOutput | raw HTML |
| imageOutput | image |
| plotOutput | plot |
| tableOutput | table |
| textOutput | text |
| uiOutput | raw HTML |
| verbatimTextOutput | Text |

|  |  |
| --- | --- |
| renderDataTable | DataTable |
| renderImage | images (saved as a link to a source file) |
| renderPlot | plots |
| renderPrint | any printed output |
| renderTable | data frame, matrix, other table like structures |
| renderText | character strings |
| renderUI | a Shiny tag object or HTML |

8. Reactive:

dataInput <- reactive({

getSymbols(input$symb, src = "yahoo",

from = input$dates[1],

to = input$dates[2],

auto.assign = FALSE)

})