|  |
| --- |
| ***RV1 - ui.R (Mark*** [***L***[***eonawicz***](https://github.com/leonawicz)](https://github.com/leonawicz)***)*** |
| library(shiny)  shinyUI(pageWithSidebar(          headerPanel(                  HTML(  '<div id="stats\_header">  Distributions of Random Variables  <a href="http://snap.uaf.edu" target="\_blank">  <img id="stats\_logo" align="right" alt="SNAP Logo" src="http://www.snap.uaf.edu/images/snap\_acronym\_rgb.gif" />  </a>  </div>'                  ),                  "Distributions of Random Variables"          ),          sidebarPanel(                  radioButtons("dist","Distribution type:",                          list("Normal"**=**"norm",  "Uniform"**=**"unif",  "t"**=**"t","F"**=**"F",  "Gamma"**=**"gam",  "Exponential"**=**"exp",  "Chi-square"**=**"chisq",  "Log-normal"**=**"lnorm",  "Beta"**=**"beta")),                  sliderInput("n","Sample size:",1,1000,500),  **# Conditional Panel – depends on which distribution chosen**                  uiOutput("dist1"),                  uiOutput("dist2"),                  checkboxInput("density","Show density curve",**FALSE**),  **# Conditional Panel – density required?**    conditionalPanel(                          condition**=**"input.density==true",                          numericInput("bw","bandwidth:",1)                  ),                  downloadButton('dldat', 'Download Sample')          ),  # ***RV1 - ui.R (Mark*** [***L***[***eonawicz***](https://github.com/leonawicz)](https://github.com/leonawicz)***)***  (Continued)  **# Main Panel – three Tabs – three inputs : plot summary table**    mainPanel(            tabsetPanel(                  tabPanel("Plot",plotOutput("plot",height**=**"600px")),                  tabPanel("Summary",verbatimTextOutput("summary")),                  tabPanel("Table",tableOutput("table"))                  )          )  )) |

|  |
| --- |
| ***RV1 - server.R (Mark*** [***L***[***eonawicz***](https://github.com/leonawicz)](https://github.com/leonawicz)***)*** |
| library(shiny)  library(datasets)  **# Preprocessing**  rt2 **<-** **function**(n**=**500,dft**=**15){ rt(n**=**n,df**=**dft) }  formals(rgamma)[1**:**2] **<-** c(500,1)  rchisq2 **<-** **function**(n**=**500,dfx**=**1){ rchisq(n**=**n,df**=**dfx) }  formals(rf)[1**:**3] **<-** c(500,1,15)  rexp2 **<-** **function**(n**=**500,rate2**=**1){ rexp(n**=**n,rate**=**rate2) }  formals(rbeta)[1**:**3] **<-** c(500,2,2)  shinyServer(**function**(input,output){          dat **<-** reactive({                  dist **<-** **switch**(input**$**dist,  **norm=rnorm,**  **unif=runif,**  **t=rt2, F=rf,**  **gam=rgamma,**  **exp=rexp2,**  **chisq=rchisq2,**  **lnorm=rlnorm,**  **beta=rbeta**)  **# Arguments for each distribution**                  def.args **<-** switch(input**$**dist,  **norm=c(input$mean,input$sd),**  **unif=c(input$min,input$max),**  **t=c(input$dft),**  **F=c(input$df1,input$df2),**  **gam=c(input$shape,input$rate),**  **exp=c(input$rate2),**  **chisq=c(input$dfx),**  **lnorm=c(input$meanlog,input$sdlog),**  **beta=c(input$shape1,input$shape2))**  f **<-** formals(dist);  f **<-** f[names(f)**!=**"n"];  len **<-** min(length(f),3-1);  f **<-** f[1**:**len]  argList **<-** list(n**=**input**$**n)  **for**(i **in** 1**:**len) {  argList[[names(f)[i]]] **<-** def.args[i]  }  **return**(list(do.call(dist,argList),names(f)))          })  **#Outputs – output sent back to SidePanel**          output**$**dist1 **<-** renderUI({                  lab **<-** **switch**(input**$**dist,  norm**=**"Mean:",  unif**=**"Minimum:",  t**=**"Degrees of freedom:",  F**=**"Numerator degrees of freedom:",  gam**=**"Shape:",  exp**=**"Rate:",  chisq**=**"Degrees of freedom:",  lnorm**=**"Mean(log):",  beta**=**"Alpha:")                  ini **<-** **switch**(input**$**dist,                          norm**=**0, unif**=**0, t**=**15,  F**=**1, gam**=**1, exp**=**1,  chisq**=**1, lnorm**=**0, beta**=**2)                  numericInput(dat()[[2]][1],lab,ini)          })    **#Outputs – output sent back to SidePanel**          output**$**dist2 **<-** renderUI({                  lab **<-** **switch**(input**$**dist,                          norm**=**"Standard deviation:",  unif**=**"Maximum:",  F**=**"Denominator degrees of freedom:",  gam**=**"Rate:",  lnorm**=**"Standard deviation(log)",  beta**=**"Beta:")         ini **<-** **switch**(input**$**dist,  norm**=**1, unif**=**1,  F**=**15, gam**=**1,  lnorm**=**1, beta**=**2)  **if**(any(input**$**dist**==**c("norm","unif","F","gam","lnorm","beta"))) numericInput(dat()[[2]][2],lab,ini)          })    **#Download Handler**     output**$**dldat **<-** downloadHandler(          filename **=** **function**() { paste(input**$**dist, '.csv', sep**=**'') },          content **=** **function**(file) {          write.csv(data.frame(x**=**dat()[[1]]), file)                  }          )  #**Output Plot (Main Panel)**          output**$**plot **<-** renderPlot({                  dist **<-** input**$**dist                  n **<-** input**$**n                  hist(dat()[[1]],#  main**=**"",xlab**=**"Observations",  col**=**"orange",cex.axis**=**1.2,  cex.lab**=**1.2,prob**=**T)  **#Density Plot**  **if**(input**$**density) lines(density(dat()[[1]],adjust**=**input**$**bw),lwd**=**2)          })  **#Output Summary (Main Panel)**            output**$**summary **<-** renderPrint({                  summary(dat()[[1]])          })  **#Output Table (Main Panel)**            output**$**table **<-** renderTable({                  data.frame(x**=**dat()[[1]])          })  }) |