user interface

- User interface layout and components
- Adding formatted text
- Adding an image

user interface

```
ui <- fluidPage(

  # Application title
  titlePanel("Old Faithful Geyser Data"),

  # Sidebar with a slider input for number of bins
  sidebarLayout(
    sidebarPanel(
      sliderInput("bins",
                  "Number of bins:",
                  min = 1,
                  max = 50,
                  value = 30)
    ),

    # Show a plot of the generated distribution
    mainPanel(
      plotOutput("distPlot")
    )
  )
)
```

user interface: fluidPage()

```
# Define UI for application that draws a histogram
ui <- fluidPage(

  rest of code for titlePanel

  rest of code for sliderInput

  rest of code for displaying histogram


)
```

fluidPage() function

display will change according to the size of your window (i.e. it's fluid)

user interface: titlePanel()

```
ui <- fluidPage(

  # Application title
  titlePanel("Old Faithful Geyser Data"),

  # Sidebar with a slider input for number of bins
  sidebarLayout(
    sidebarPanel(
      sliderInput("bins",
                  "Number of bins:",
                  min = 1,
                  max = 50,
                  value = 30)
    ),

    # Show a plot of the generated distribution
    mainPanel(
      plotOutput("distPlot")
    )
  )

)
```

titlePanel() function

Application title, and by default also the name of the browser window.

Note the comma at the end of the function, and for the other functions that follow (except the very last one).

i.e. fluidPage(titlePanel, sidebarLayout)

user interface: sidebarLayout()

```
ui <- fluidPage(

  # Application title
  titlePanel("Old Faithful Geyser Data"),

  # Sidebar with a slider input for number of bins
  sidebarLayout(

    sidebarPanel(
      sliderInput("bins",
                  "Number of bins:",
                  min = 1,
                  max = 50,
                  value = 30)
    ),

    # Show a plot of the generated distribution
    mainPanel(
      plotOutput("distPlot")
    )

  )

)
```

sidebarLayout() function

Set up a sidebar layout:

sidebarLayout(sidebarPanel, mainPanel)

By default the sidebarPanel, with the slider input takes up 1/3 of the width.

By default the mainPanel, with the plot output takes up 2/3 of the width

For both the sidebarPanel and the mainPanel can put more than one element

user interface exercise: histogram with extras

(1) Copy and paste the **Default** directory, so that you have another sub-directory under **workshop apps**.

(2) Rename the new sub-directory **Hist-With-Extras** (or similar)

(3) Go into **Hist-With-Extras** and double-click on app.R

user interface: add a dummy slider

```
# Add in a dummy slider for the side bar panel
# Make sure there's a comma after the first slider input function!
# Changes: "bins" to "bins2"  (so that it does absolutely nothing)

sidebarPanel(

   sliderInput("bins",
               "Number of bins:",
               min = 1,
               max = 50,
               value = 30),

    sliderInput("bins2",
               "Number of bins dummy input:",
               min = 10,
               max = 40,
               value = 20)
),
```

user interface exercise: add another plot

Exercise. Add in a second histogram below the first, with red coloured bars.

Hint. You'll need to generate the second histogram in the server part of the code, then display it in the user interface.

The server code is just a bunch of R code, so need to put a comma at the end/between sections of code.

user interface exercise: add another plot

Exercise. Add in a second histogram below the first, with red coloured bars.

```
ui <- fluidPage(

…

    # Show a plot of the generated distributions
     mainPanel(
       plotOutput("distPlot"),
       plotOutput("distPlot2")
     )

)

server <- function(input, output) {

 output$distPlot2 <- renderPlot({
      # generate bins based on input$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 = 'red', border = 'white')
   })

}
```

user interface : more advanced layout

I’ve shown how a basic layout works. For layouts with more menu levels the **shinydashboard** library streamlines the coding

https://rstudio.github.io/shinydashboard/

To find out more about how layouts work check out

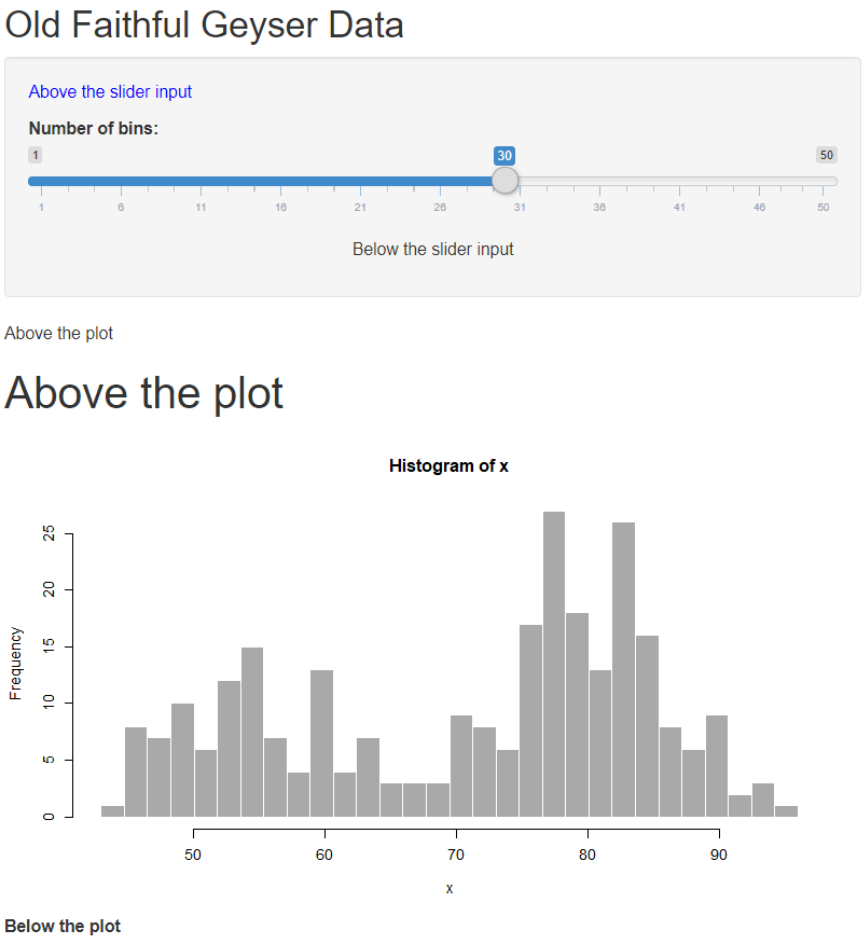
https://shiny.rstudio.com/articles/layout-guide.html

user interface: adding formatted text

Extra text in orange

```
sidebarLayout(
    sidebarPanel(
      p("Above the slider input“, style = “color:blue”),
      sliderInput("bins",
                  "Number of bins:",
                  min = 1,
                  max = 50,
                  value = 30),
      p("Below the slider input“, align = “center”)
    ),

    # Show a plot of the generated distribution
    mainPanel(
      p(“Above the plot"),
      h1(“Above the plot"),
      plotOutput("distPlot"),
      strong(“Below the plot")
    )
  )
```



user interface: adding formatted text

Shiny HTML tag functions. Some more, go crazy and have a play!

https://shiny.rstudio.com/tutorial/written-tutorial/lesson2/

https://www.w3schools.com/tags/tag_hn.asp

shiny function	HTML5 equivalent	creates
`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`	` `	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

user interface: adding an image

To do this

(1) use the **img** function

(2) make a directory “www” in the same place as your app.R code

(3) put the image in the “www” directory

Then in the user interface put somewhere inside the fluidPage() function

img(src = “myimage.png”, height = 100, width = 300)

* Can put CSS (Cascading Style Sheets) files in the “www” directory → define general formatting features of a web page (e.g. layout, colors, fonts)

user interface: adding an image

Code and app output snippets

```
# Sidebar with a slider input for number of bins
  sidebarLayout(
    sidebarPanel(

      sliderInput("bins",
                  "Number of bins:",
                  min = 1,
                  max = 50,
                  value = 30),

      img(src = "screen-grab.png", height = 100, width = 300),

    ),
```