

Shiny apps

- creates a UI
- web based
- uses R (and HTML/CSS)

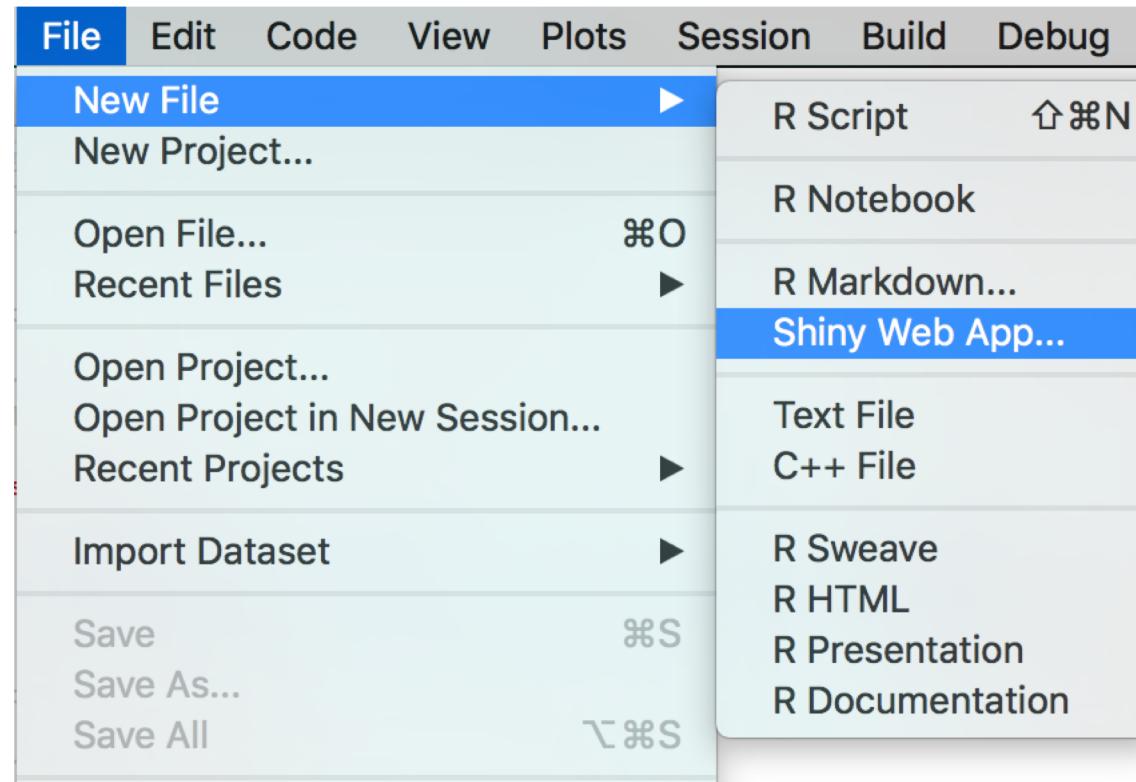
Examples

- [BoxPlotR](#)
- [PlotsOfData](#)
- [Plasmid database](#)

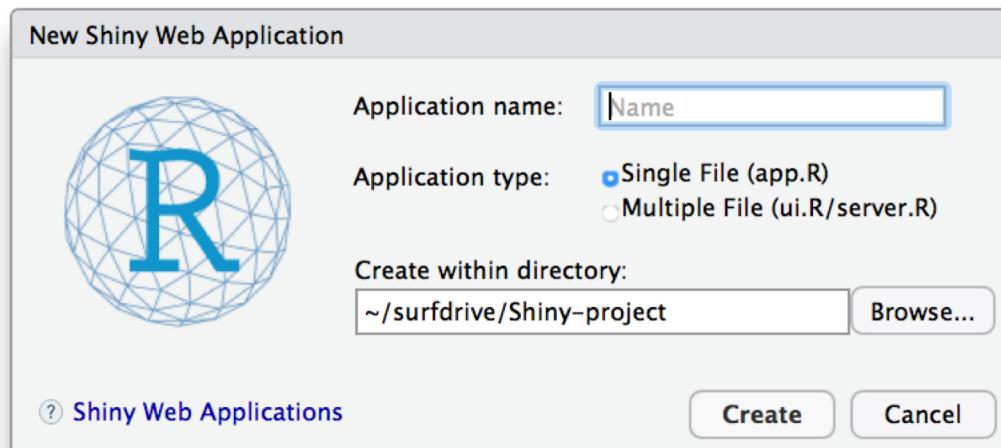
Why?

- Online (and offline) access
- Open source
- User friendly

First steps - RStudio



First steps - RStudio



The screenshot shows the RStudio interface with the following components:

- Top Bar:** Contains the RStudio logo, file menu (File, Edit, View, Project, etc.), Go to file/func, Addins dropdown, and a Project: (None) dropdown.
- Left Panel (Script Editor):** Displays the content of the `app.R` script. The code defines a Shiny web application for visualizing Old Faithful Geyser Data. It includes a sidebar with a slider input for the number of bins (ranging from 1 to 50, with a default of 30), and a main panel containing a histogram output named `distPlot`. The server logic generates bins based on the input and draws a histogram with the specified number of bins.
- Middle Panel (Toolbar):** Features a prominent red circle around the **Run App** button, which is used to execute the Shiny application defined in the script.
- Right Panel (Environment):** Shows the Global Environment tab with the message "Environment is empty".
- Bottom Panel (Console):** Displays the R console history with entries like >, >, >, >, >.

```
1 #
2 # This is a Shiny web application. You can run the application by clicking
3 # the 'Run App' button above.
4 #
5 # Find out more about building applications with Shiny here:
6 #
7 #   http://shiny.rstudio.com/
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14
15   # Application title
16   titlePanel("Old Faithful Geyser Data"),
17
18   # Sidebar with a slider input for number of bins
19   sidebarLayout(
20     sidebarPanel(
21       sliderInput("bins",
22                   "Number of bins:",
23                   min = 1,
24                   max = 50,
25                   value = 30)
26     ),
27
28     # Show a plot of the generated distribution
29     mainPanel(
30       plotOutput("distPlot")
31     )
32   )
33 )
34
35 # Define server logic required to draw a histogram
36 server <- function(input, output) {
37
38   output$distPlot <- renderPlot({
39     # generate bins based on input$bins from ui.R
40     x   <- faithful[, 2]
41     bins <- seq(min(x), max(x), length.out = input$bins + 1)
42
43     # draw the histogram with the specified number of bins
44     hist(x, breaks = bins, col = 'darkgray', border = 'white')
45   })
46 }
47
48 # Run the application
49 shinyApp(ui = ui, server = server)
50
```

```
20 sidebarPanel(  
21   sliderInput("bins",  
22     "Number of bins:",  
23     min = 1,  
24     max = 50,  
25     value = 30)
```

```
39 # generate bins based on input$bins from ui.R  
40 x <- faithful[, 2]  
41 bins <- seq(min(x), max(x), length.out = input$bins - 1)  
42  
43 # draw the histogram with the specified number of bins  
44 hist(x, breaks = bins, col = 'darkgray', border = 'white')
```

30

```
plotOutput("distPlot")
```

38 -

```
output$distPlot <- renderPlot({
```

Building an app: step-by-step

https://github.com/JoachimGoedhart/Shiny_step-by-step

Running your app

- Via web: Shinyapps.io
- Via web: Shinyserver
- Local: needs R/Rstudio

Limitations

- Upload limit (5 Mb?)
- Connection loss
- Preferences/data not stored