

Plotting Part 2.0

Shiny

The R Bootcamp
Twitter: [@therbootcamp](#)

September 2017

Shiny

*What is the Matrix
(**Shiny**)?*



*No one can be told what
the Matrix (**Shiny**) is...
You have to see it for
yourself.*



This is Shiny

[Shiny \(https://shiny.rstudio.com/\)](https://shiny.rstudio.com/)

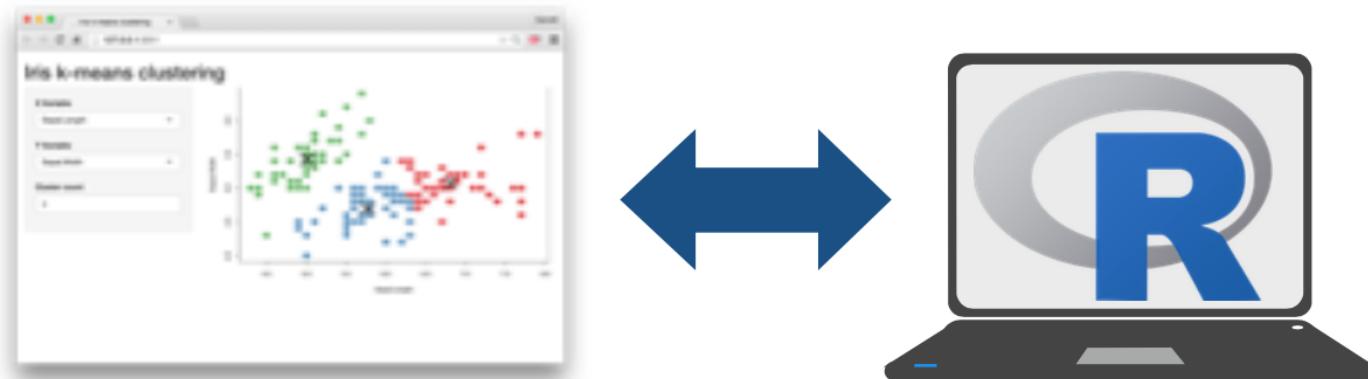
from [\(https://www.rstudio.com/\)](https://www.rstudio.com/)

[Back to Gallery \(https://shiny.rstudio.com/gallery/\)](https://shiny.rstudio.com/gallery/)

[Get Code \(https://github.com/rstudio/shiny-e:](https://github.com/rstudio/shiny-examples)

What is Shiny?

A **Shiny** app is a web page (**UI**) connected to a computer running a live R session (**Server**)



Users can manipulate the UI, which will cause the server to update the UI's displays (by running R code).

R Studio

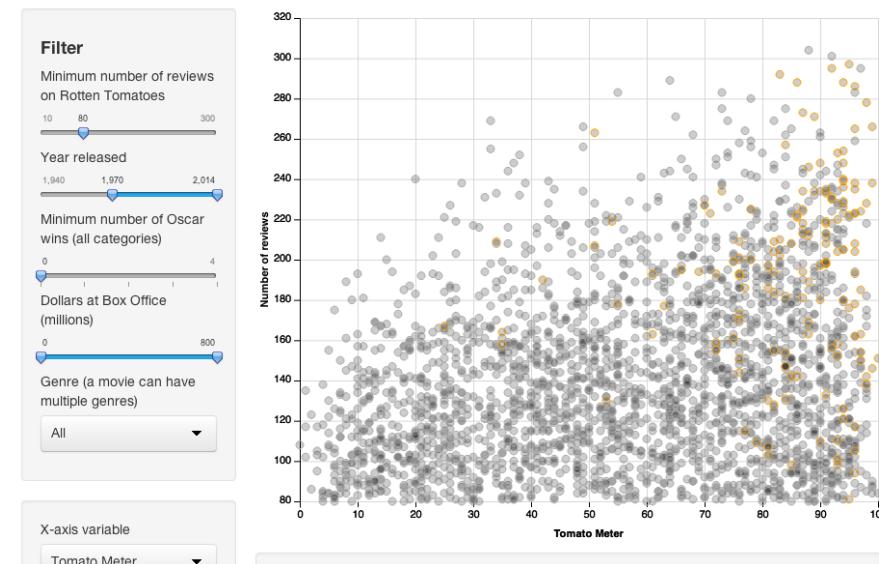
What does that mean?

- Shiny allows you to easily create interactive websites to explore, analyse, and visualize data
- No need to learn HTML, CSS, Javascript



I can easily do this all in Shiny!

Movie explorer



Histogram Example

Balloon Analogue Risk Task

Please Wait



FFTrees Example

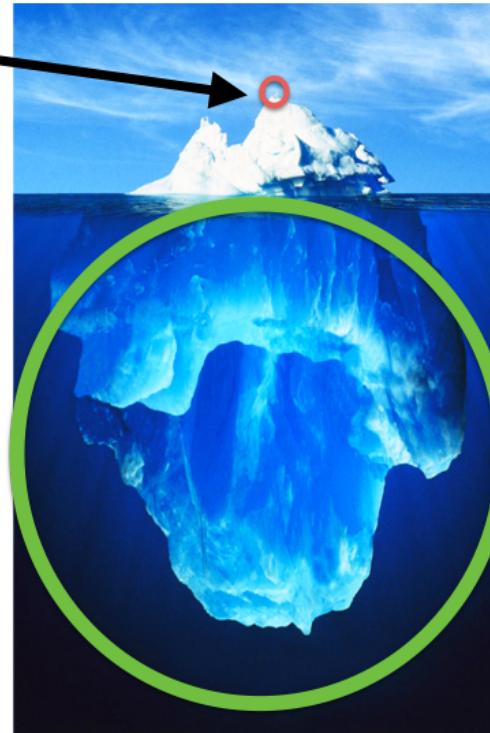
Please Wait



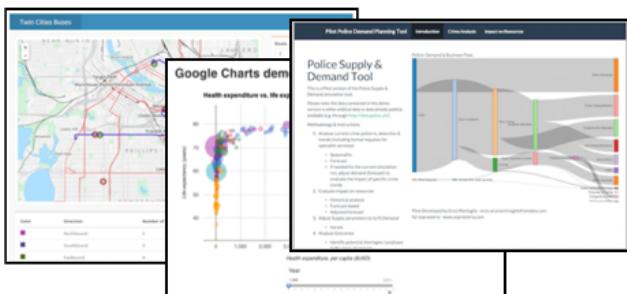
GlioVis

Shiny

What we will cover today



What you can DO with shiny



R Studio

Shiny Tutorials

R Studio has great tutorials for creating Shiny Apps

Shiny from R Studio

Learn Shiny

The tutorials on this page are primarily designed for users who are new to Shiny and want a guided introduction. If you use Shiny on a regular basis, you may want to skip these tutorials and visit the [articles](#) section where we cover individual Shiny topics at an advanced level.

Get started materials are organized in two sections: [videos](#) and [written](#) tutorials.

Video tutorials

How to Start Shiny tutorial

The How to Start Shiny video series will take you from R programmer to Shiny developer. Watch the complete tutorial, or jump to a specific chapter by clicking a link below. The entire tutorial is two hours and 25 minutes long.

A screenshot of a video player interface. At the top, it says 'R Studio' and provides contact information: '250 Northern Ave, Boston, MA 02210', 'Phone: 844-448-3212', 'Email: info@rstudio.com', and 'Web: https://rstudio.com'. Below this, the title 'How to start with Shiny, Part 1' is displayed in large bold letters. Underneath the title, the subtitle 'How to build a Shiny App' is shown. A dropdown menu labeled 'Choice 1' is open, showing three options: 'Choice 1' (selected), 'Choice 2', and 'Choice 3'.

R Studio

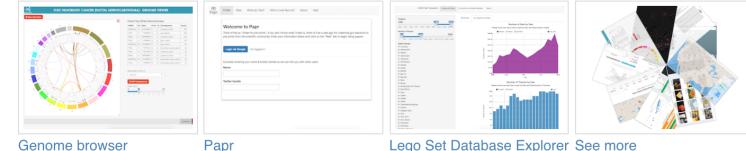
Learn by example. Tons of shiny apps online. Most code is available.

Shiny from R Studio

Gallery

Shiny User Showcase

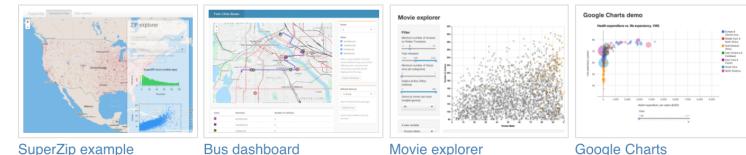
The [Shiny User Showcase](#) contains an inspiring set of sophisticated apps developed and contributed by Shiny users.



Genome browser Papr Lego Set Database Explorer See more

Interactive visualizations

Shiny is designed for fully interactive visualization, using JavaScript libraries like d3, Leaflet, and Google Charts.

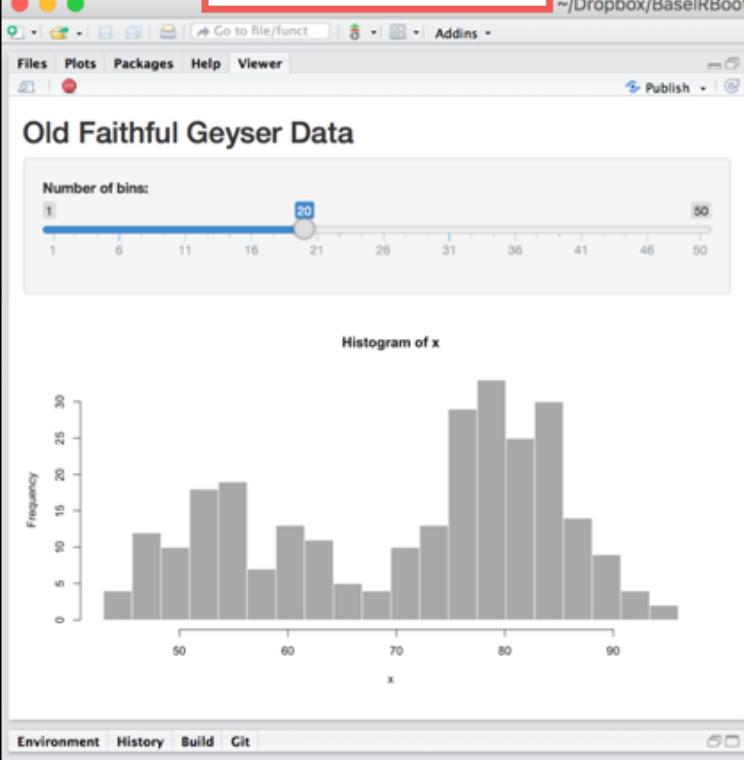


R Studio

How programming a Shiny App looks

App Preview

Shiny App Code



```
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     mainPanel(  
27       histogramOutput("distPlot"))  
28   ))  
29  
30 # Compute and display summary statistics on the data  
31 summaryData <- reactive({  
32   stats <-  
33     list(n = nrow(faithful),  
34           mean = mean(faithful$waiting),  
35           sd = sd(faithful$waiting),  
36           min = min(faithful$waiting),  
37           q1 = quantile(faithful$waiting, 0.25),  
38           median = median(faithful$waiting),  
39           q3 = quantile(faithful$waiting, 0.75),  
40           max = max(faithful$waiting))  
41   return(stats)  
42 })  
43  
44 # Define analysis function  
45 distPlot <- function(input, output) {  
46   # Create a histogram  
47   output$distPlot <- renderHistogram(function() {  
48     hist(faithful$waiting, breaks = input$bins, xlab = "Waiting Time (min)",  
49          ylab = "Frequency", main = "Histogram of Waiting Time")  
50   })  
51 }  
52  
53 # Run the application  
54 # ----  
55 shinyApp(ui = ui, server = server)
```

Console is active

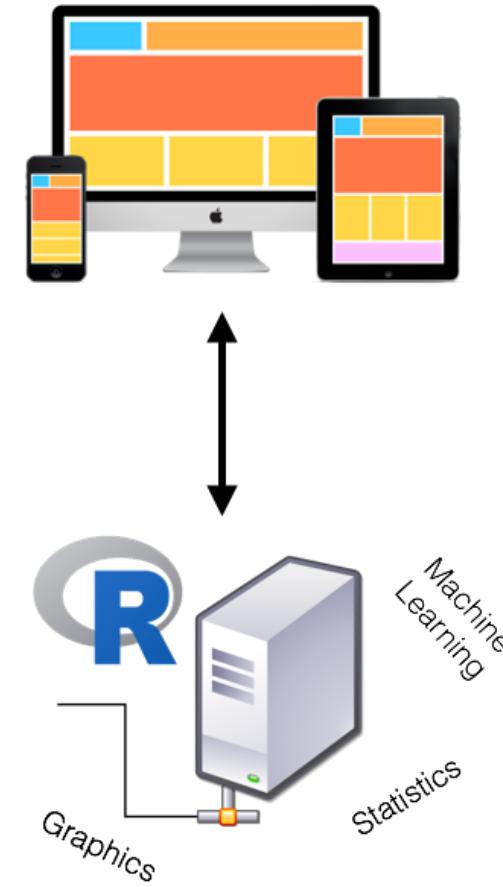
Structure of a Shiny App

User Interface: **ui()**

```
ui <- (
  # Set overall layout
  # Add widgets,
  # Display plots
)
```

R Server: **server()**

```
server <- function(input, output) {
  # Access Data
  # Run analyses
  # Create and render plots
}
}
```



Let's explore the user interface of an app!

P.S. You'll create this app in the practical!

User Interface

The final app!

Please Wait



User Interface

The user interface typically contains two main components: Widgets and

User Interface, Widgets

- Widgets are simple fields added to the user interface for users to add inputs.

User Interface, Layout

- You can control the layouts of apps with layout functions

Server

- All R code that creates plots, does machine learning, accesses databases, searches twitter data (really anything!) goes in the `server()` function.

Server

How does the server communicate with the user interface?

To send output to the user interface, you must use a render function

- To present output (e.g.; a plot), it must be *rendered* in the server using a special rendering function `renderXX()`,
- Once it is rendered, it is sent to the output and displayed in the user interface using an `xxOutput()` function.

Server

Please Wait

...

```
library(shiny)

# User Interface:
ui <- fluidPage(
  mainPanel(
   textInput("Title", "Title"),
    plotOutput("myplot") # Create the plot
  )
)

server <- function(input, output) {

  # Define x
  x <- ChickWeight$weight

  # Send rendered plot to output
  output$myplot <- renderPlot({
    hist(x, main = input>Title)
  })
}

shinyApp(ui = ui, server = server)
```

Rendering output

The Shiny cheatsheet explains the most common functions for rendering and presenting output.

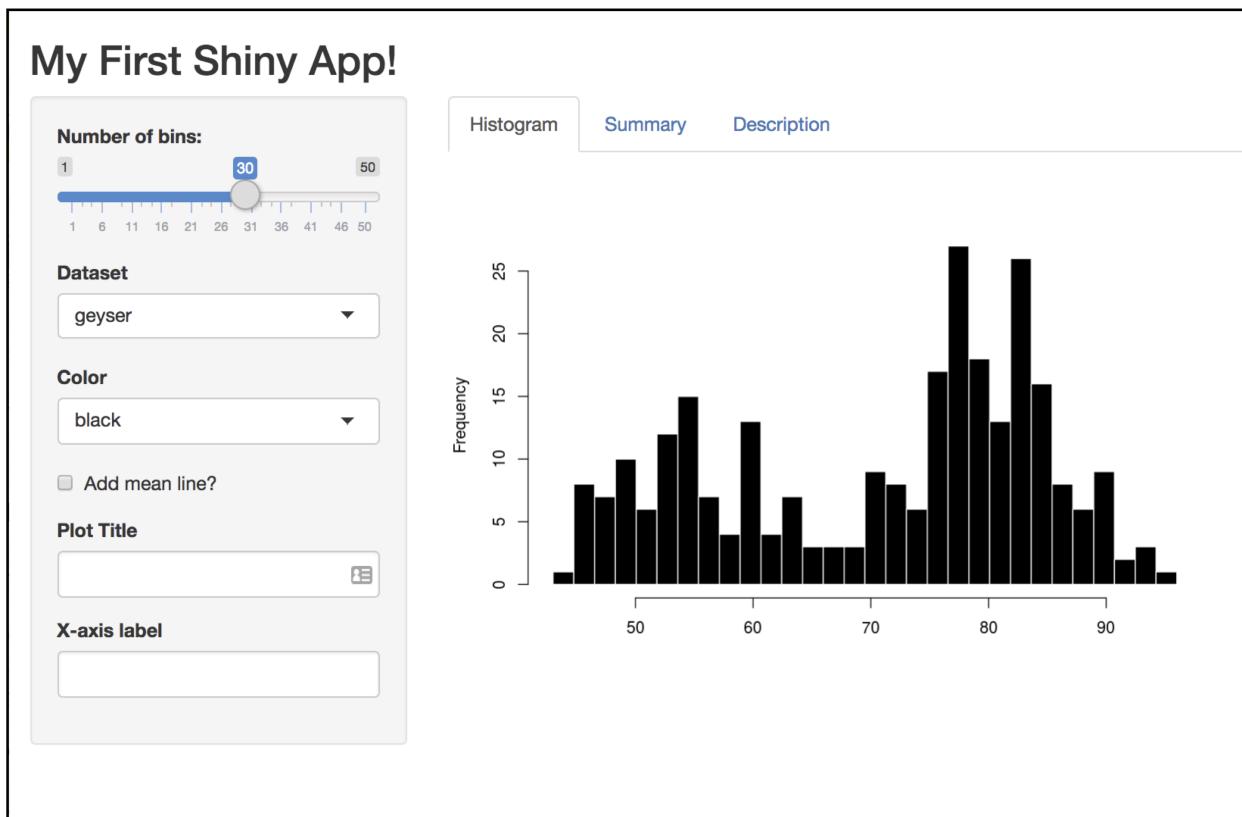
Publishing (hosting) an app

R Studio

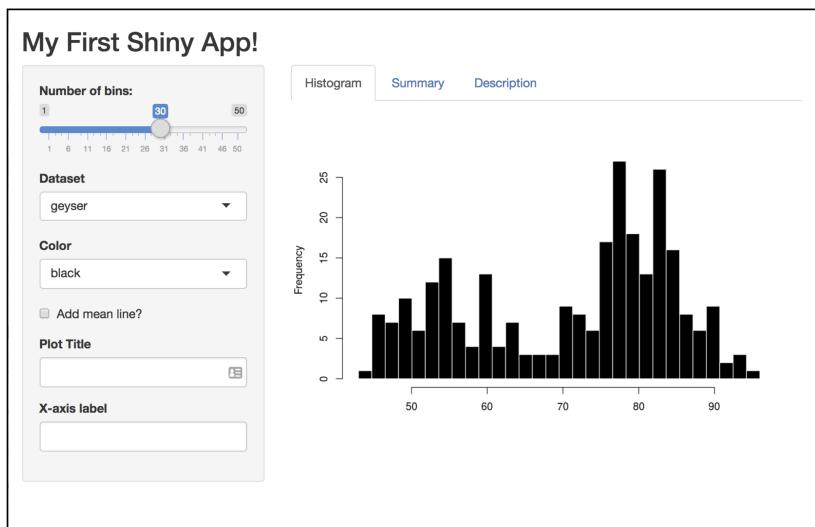
- You can always run a Shiny app locally on your machine.
- To get it online, you need to put it on a Shiny Server.
- Publish an app (with some restrictions) at <http://shinyapps.io> from RStudio with one click!
- Install a local server at your business (RStudio: \$10,000 / year)
- Other providers exist (e.g.; Amazon Web Services)

Practical

- In the practical, we will create the following app from scratch, and publish it online!



Questions?



Shiny from R Studio

Gallery

Shiny User Showcase

The [Shiny User Showcase](#) contains an inspiring set of sophisticated apps developed and contributed by Shiny users.



Genome browser



Pap



Lego Set Database Explorer [See more](#)



Interactive visualizations

Shiny is designed for fully interactive visualization, using JavaScript libraries like d3, Leaflet, and Google Charts.



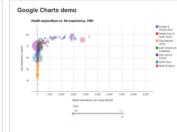
SuperZip example



Bus dashboard



Movie explorer



Google Charts

R Studio

Plotting II Practical

[Link to Plotting II practical](#)

