DSBA 5122: Visual Analytics

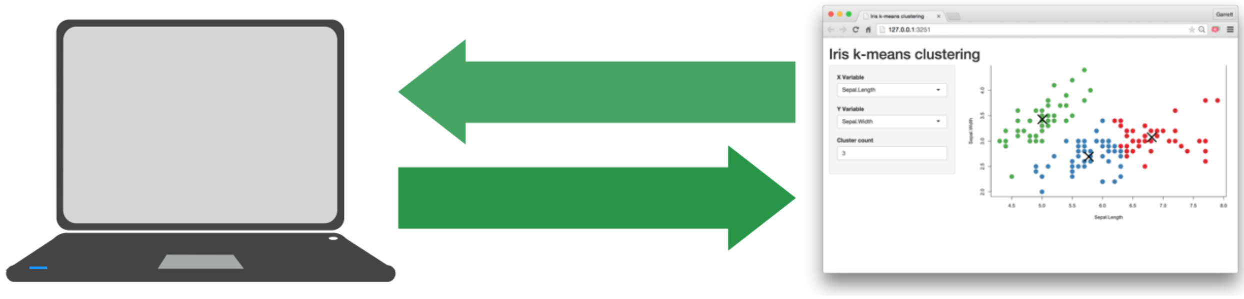
Class 8 Lab: Intro to Shiny

Ryan Wesslen

March 18, 2019

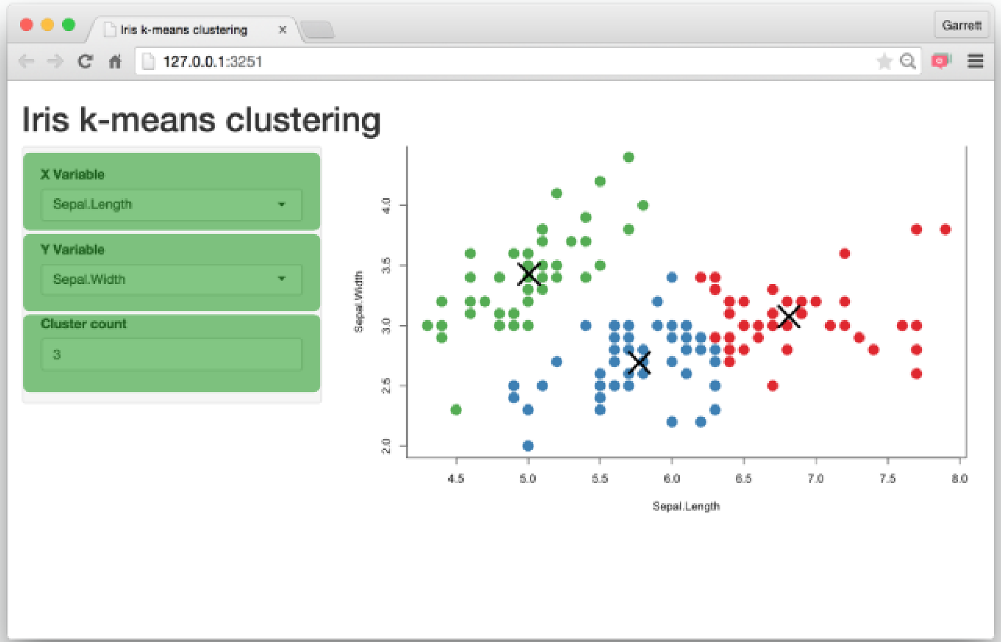
Shiny

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).

Build your app around **inputs** and **outputs**



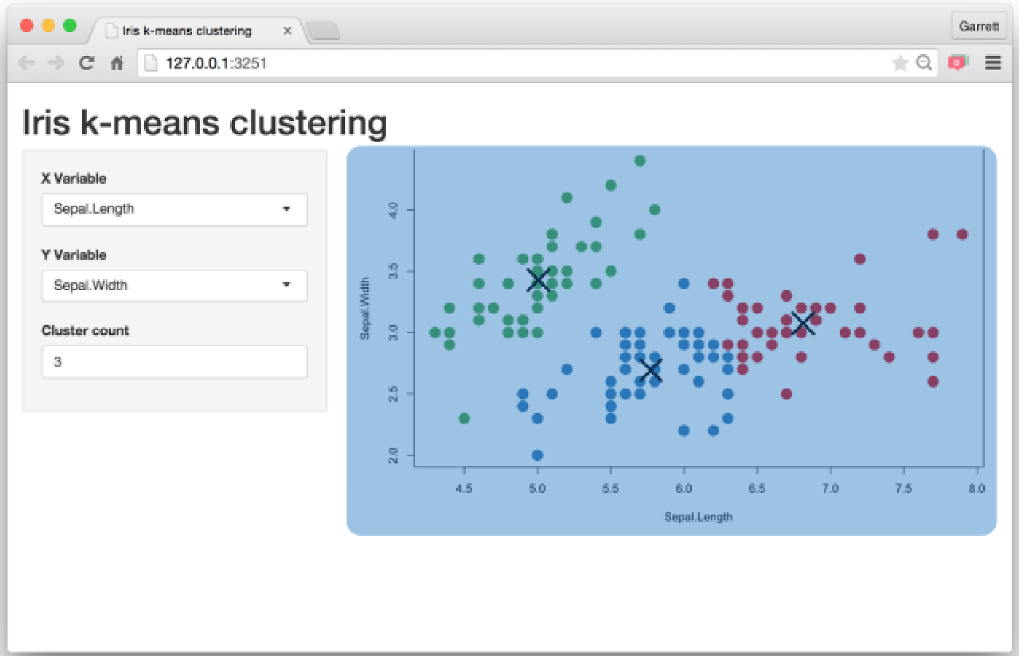
Iris k-means clustering

X Variable
Sepal.Length

Y Variable
Sepal.Width

Cluster count
3

Build your app around inputs and outputs



Iris k-means clustering

X Variable

Sepal.Length

Y Variable

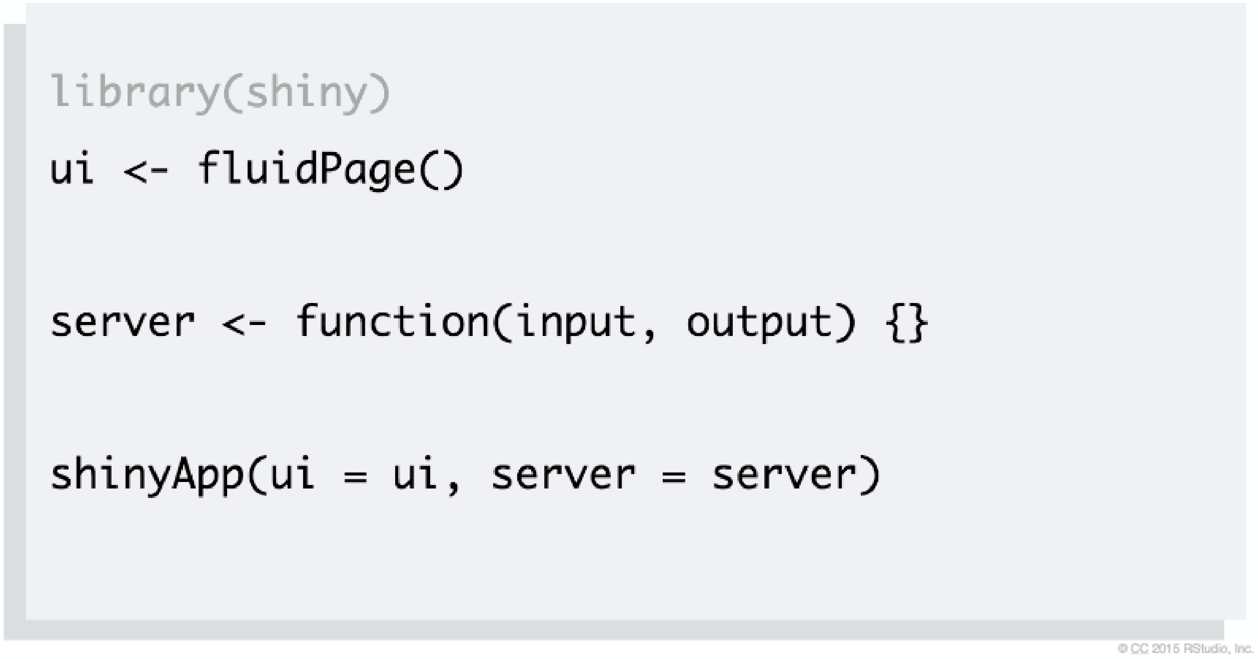
Sepal.Width

Cluster count

3

App template

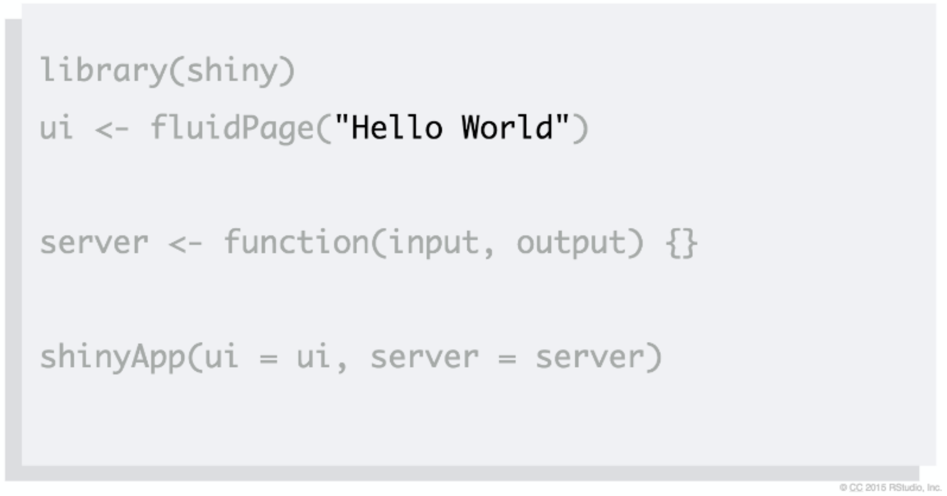
The shortest viable shiny app



```r
library(shiny)
ui <- fluidPage()


server <- function(input, output) {}


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

Try This

1. Open a new .R file
2. Type this into the file. (Do you have the shiny package?)

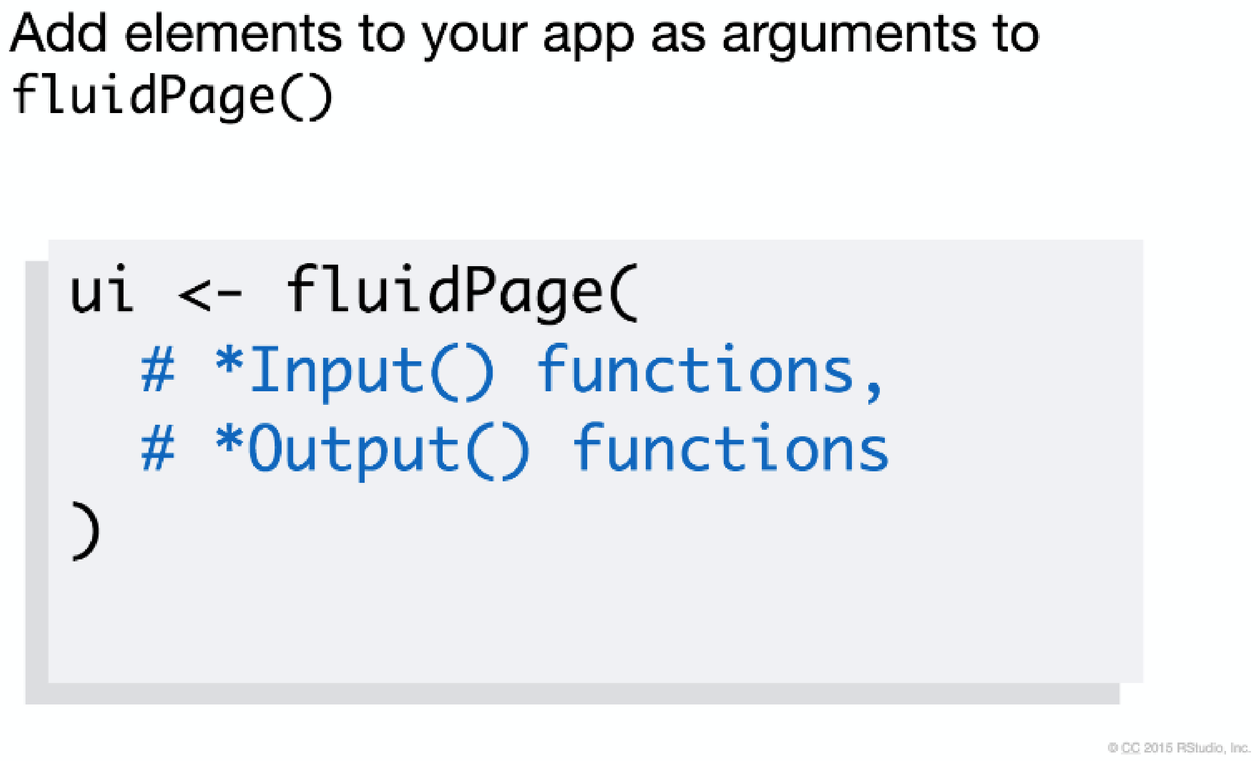


```
library(shiny)
ui <- fluidPage("Hello World")

server <- function(input, output) {}

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

3. Click "Run"

User Interface: ui() function

`fluidPage()`

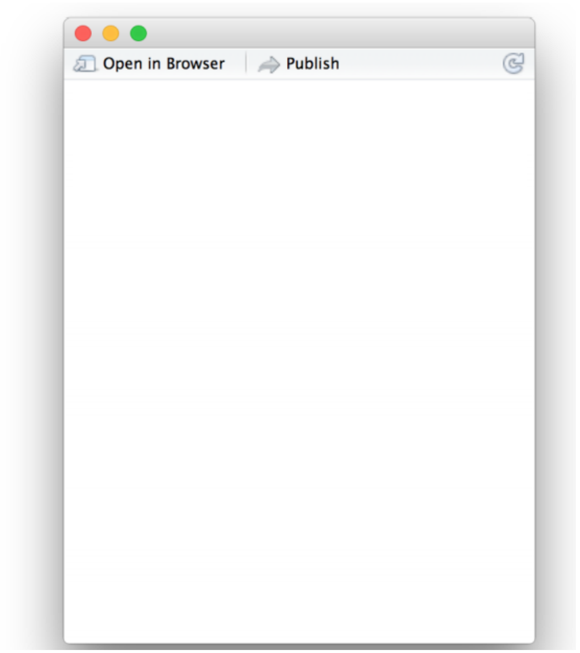


Add elements to your app as arguments to `fluidPage()`

```
ui <- fluidPage(
  # *Input() functions,
  # *Output() functions
)
```

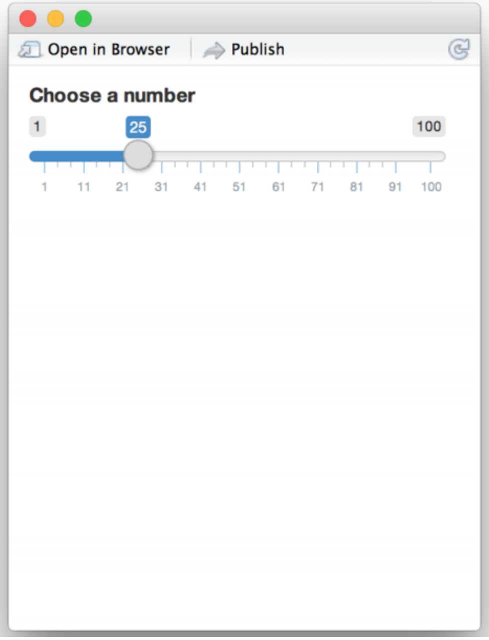
Example

```
library(shiny)
ui <- fluidPage(



)

server <- function(input, output) {}

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

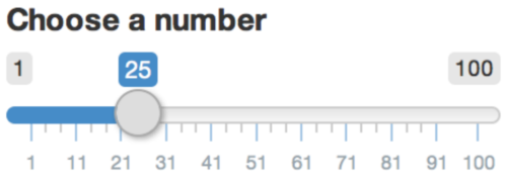


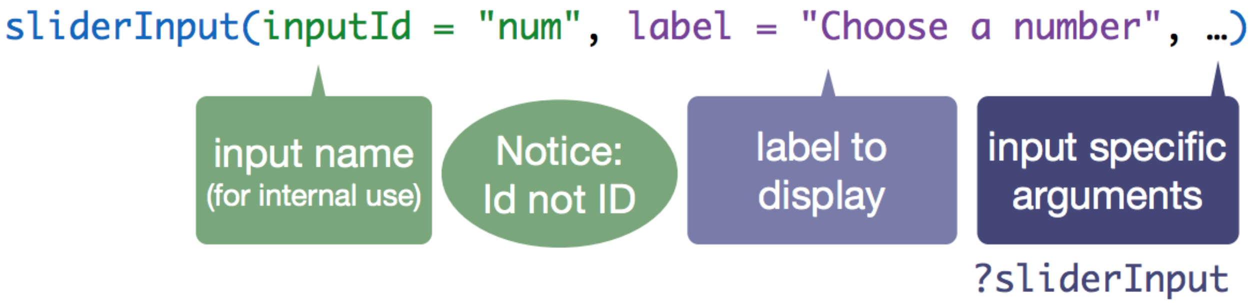
Example

```r
library(shiny)
ui <- fluidPage(
  sliderInput(inputId = "num",
    label = "Choose a number",
    value = 25, min = 1, max = 100)
)

server <- function(input, output) {}

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

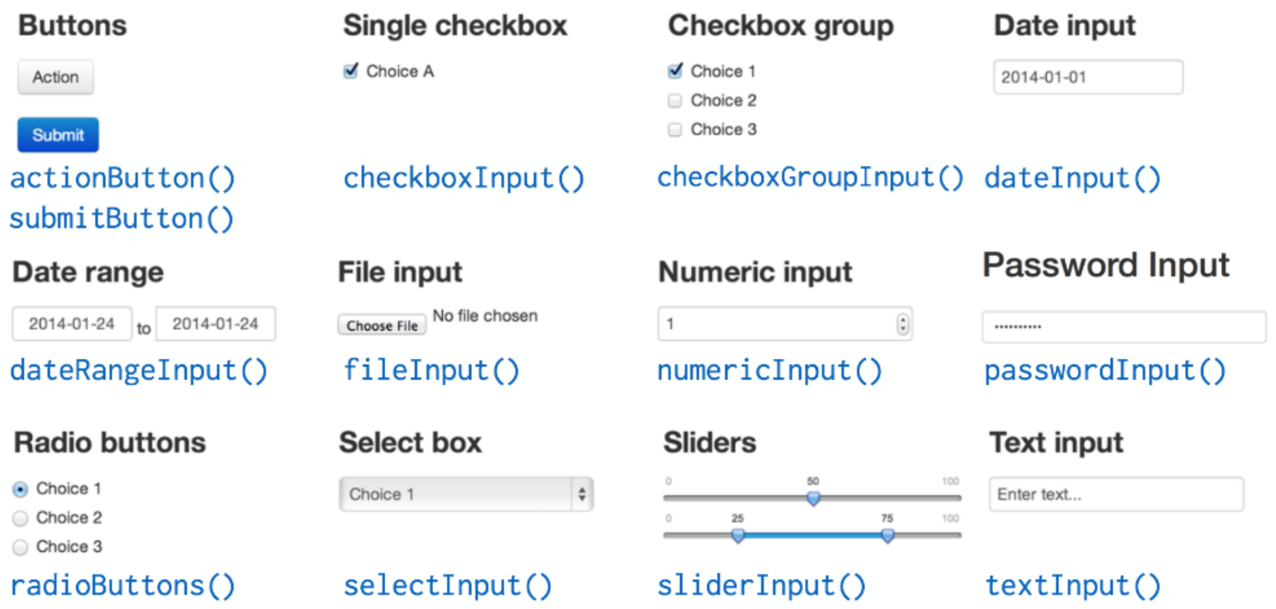


Input Syntax




```
sliderInput(inputId = "num", label = "Choose a number", …)
```

Inputs



There are multiple common **`Input()`** functions.

What's in an Input function? HTML

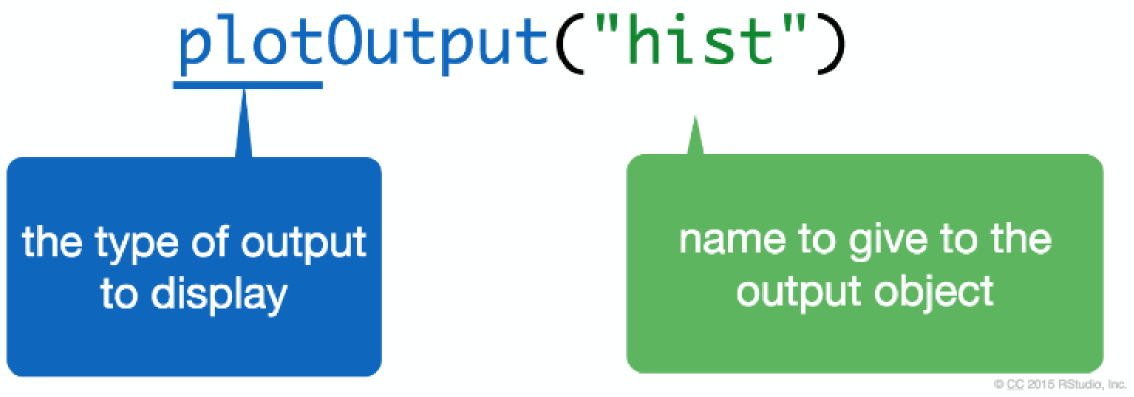
```
sliderInput(inputId = "num",
  label = "Choose a number",
  value = 25, min = 1, max = 100)
```

```
<div class="form-group shiny-input-container">
  <label class="control-label" for="num">Choose a number</label>
  <input class="js-range-slider" id="num" data-min="1" data-max="100"
   data-from="25" data-step="1" data-grid="true" data-grid-num="9.9"
   data-grid-snap="false" data-prettify-separator="," data-keyboard="true"
   data-keyboard-step="1.01010101010101"/>
</div>
```

Output Syntax

*Output()

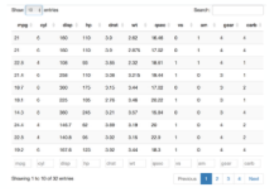
To display output, add it to `fluidPage()` with an `*Output()` function



```
plotOutput("hist")
```

the type of output to display
name to give to the output object

Output Options



Show 10 entries Search:

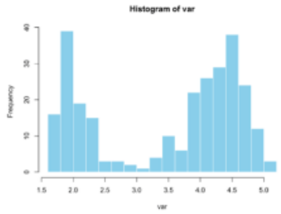
mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
21	6	160	110	3.9	2.62	16.46	0	1	4	4
21	6	160	110	3.9	2.875	17.02	0	1	4	4
22.8	4	108	93	3.85	2.32	18.61	1	1	4	1
21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
18.7	8	360	175	3.15	3.44	17.02	0	0	3	2
18.1	6	225	105	2.76	3.46	20.22	1	0	3	1
14.3	8	360	245	3.21	3.57	15.84	0	0	3	4
24.4	4	146.7	62	3.69	3.19	20	1	0	4	2
22.8	4	140.8	95	3.92	3.15	22.9	1	0	4	2
19.2	6	167.6	123	3.92	3.44	18.3	1	0	4	4

Showing 1 to 10 of 32 entries

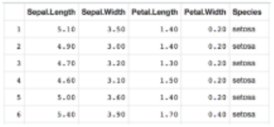
`dataTableOutput()`

`htmlOutput()`

`imageOutput()`



`plotOutput()`

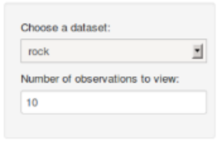


	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.10	3.50	1.40	0.20	setosa
2	4.90	3.00	1.40	0.20	setosa
3	4.70	3.20	1.30	0.20	setosa
4	4.60	3.10	1.50	0.20	setosa
5	5.00	3.60	1.40	0.20	setosa
6	5.40	3.90	1.70	0.40	setosa

`tableOutput()`

foo

`textOutput()`



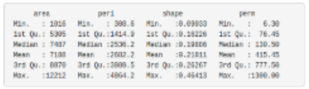
Choose a dataset:

rock

Number of observations to view:

10

`uiOutput()`

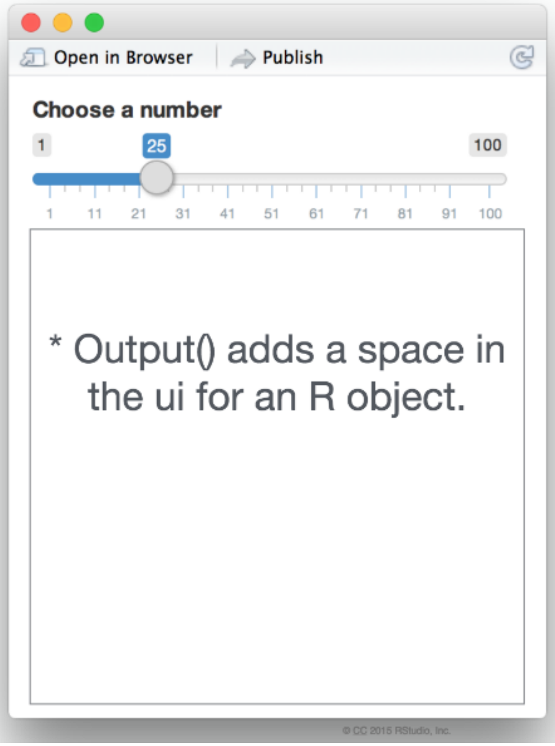


```
     area          peri           shape             perm
 Min.   : 1016   Min.   : 308.6   Min.   :0.09033   Min.   :   6.30
 1st Qu.: 5305   1st Qu.:1414.9   1st Qu.:0.16226   1st Qu.:  76.45
 Median : 7487   Median :2536.2   Median :0.19886   Median : 130.50
 Mean   : 7188   Mean   :2682.2   Mean   :0.21811   Mean   : 415.45
 3rd Qu.: 8870   3rd Qu.:3989.5   3rd Qu.:0.26267   3rd Qu.: 777.50
 Max.   :12212   Max.   :4864.2   Max.   :0.46413   Max.   :1300.00
```

`verbatimTextOutput()`

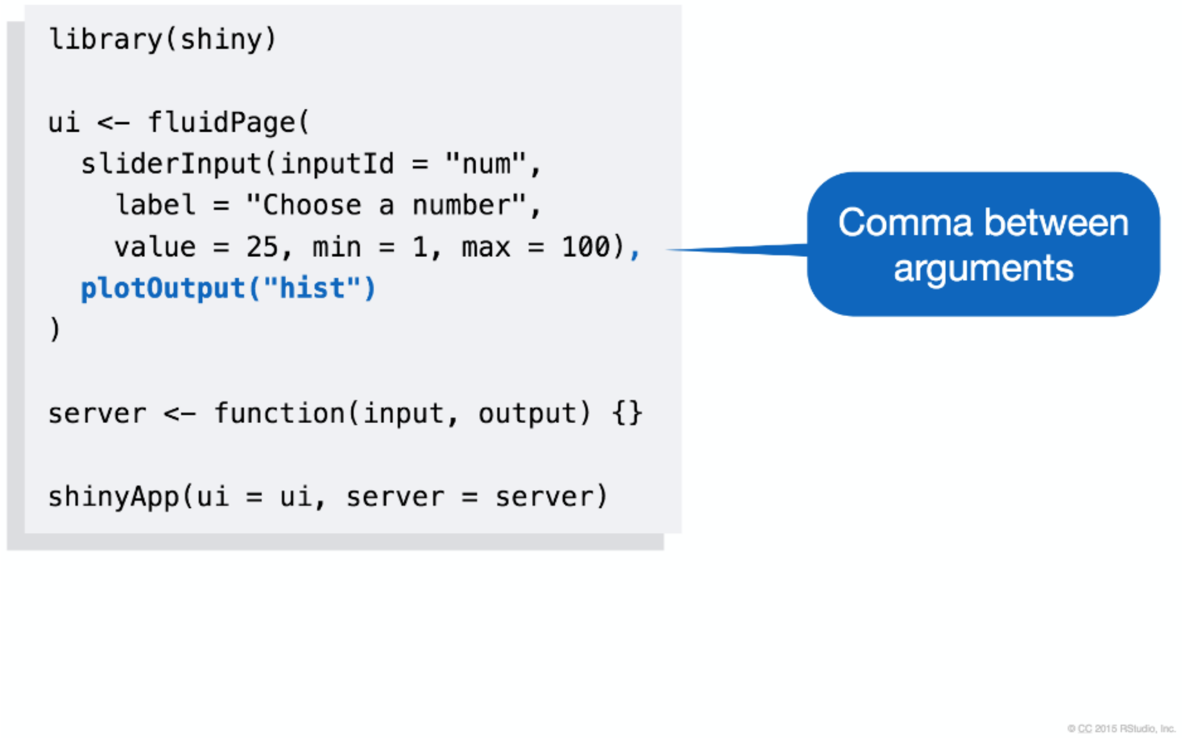
Example

```r
library(shiny)

ui <- fluidPage(
  sliderInput(inputId = "num",
    label = "Choose a number",
    value = 25, min = 1, max = 100),
  plotOutput("hist")
)

server <- function(input, output) {}

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



* Output() adds a space in the ui for an R object.

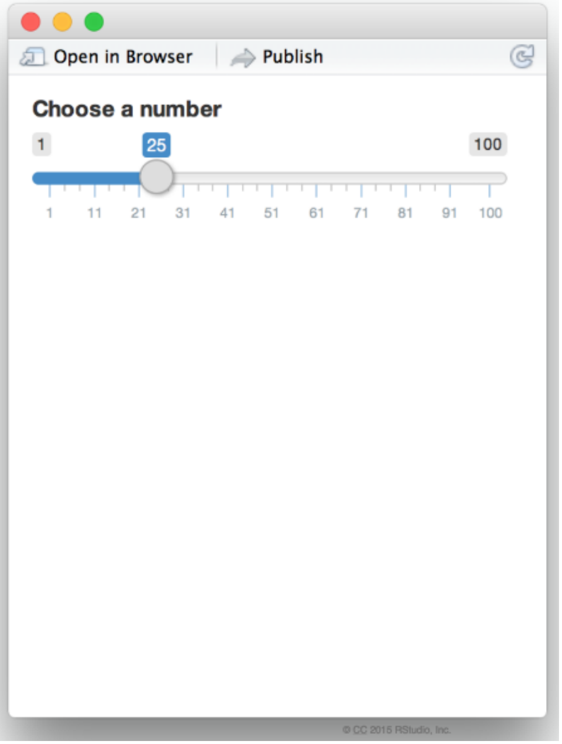
Example



```r
library(shiny)

ui <- fluidPage(
  sliderInput(inputId = "num",
    label = "Choose a number",
    value = 25, min = 1, max = 100),
  plotOutput("hist")
)

server <- function(input, output) {}

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

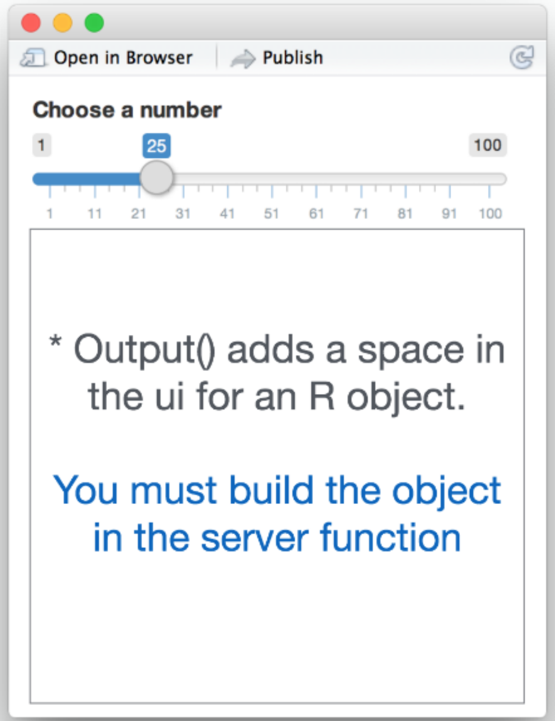
Example

```r
library(shiny)

ui <- fluidPage(
  sliderInput(inputId = "num",
    label = "Choose a number",
    value = 25, min = 1, max = 100),
  plotOutput("hist")
)

server <- function(input, output) {}

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



Example

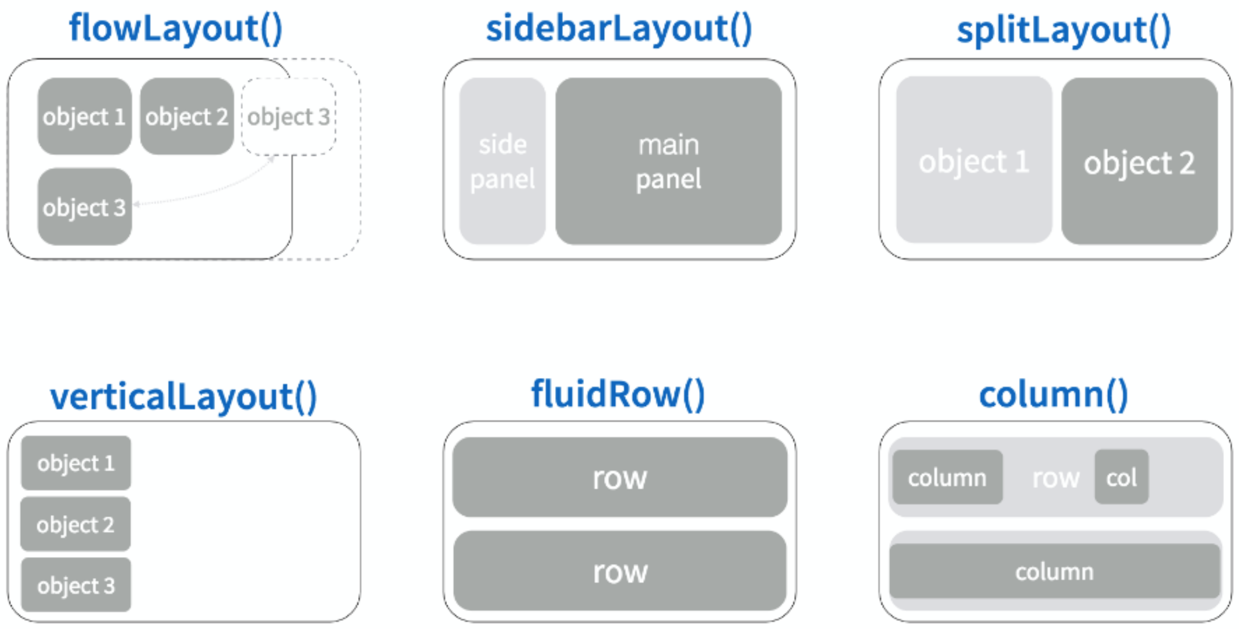
```
library(shiny)

ui <- fluidPage(
  sliderInput(inputId = "num",
    label = "Choose a number",
    value = 25, min = 1, max = 100),
  plotOutput("hist")
)

server <- function(input, output) {}

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



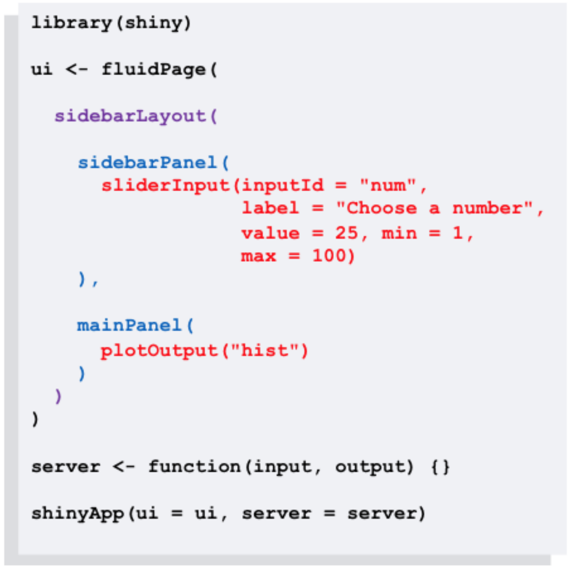
* Output() adds a space in the ui for an R object.

You must build the object in the server function

Layouts


flowLayout()
object 1
object 2
object 3
object 3
sidebarLayout()
side panel
main panel
splitLayout()
object 1
object 2
verticalLayout()
object 1
object 2
object 3
fluidRow()
row
row
column()
column
row
col
column

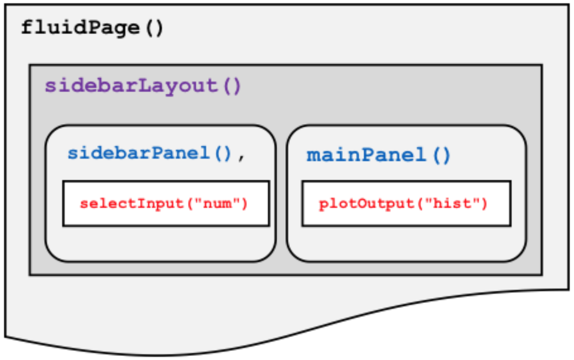
sidebarLayout()



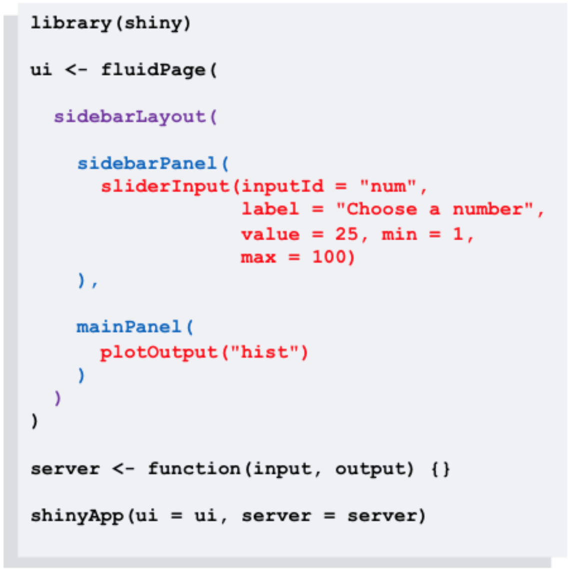
```r
library(shiny)

ui <- fluidPage(

  sidebarLayout(

    sidebarPanel(
      sliderInput(inputId = "num",
                  label = "Choose a number",
                  value = 25, min = 1,
                  max = 100)
    ),

    mainPanel(
      plotOutput("hist")
    )
  )
)

server <- function(input, output) {}

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

sidebarLayout()

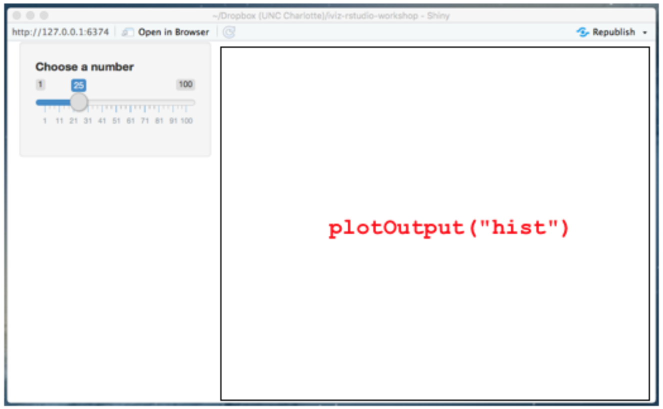
```
library(shiny)

ui <- fluidPage(

  sidebarLayout(

    sidebarPanel(
      sliderInput(inputId = "num",
                  label = "Choose a number",
                  value = 25, min = 1,
                  max = 100)
    ),

    mainPanel(
      plotOutput("hist")
    )
  )
)

server <- function(input, output) {}

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



sidebarLayout()

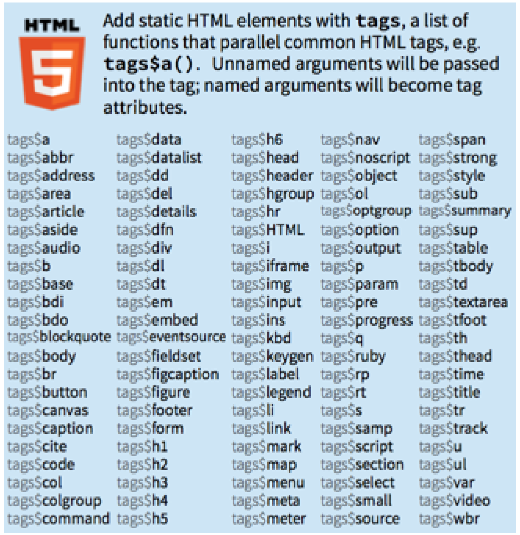


```
library(shiny)

ui <- fluidPage(

  sidebarLayout(

    sidebarPanel(
      sliderInput(inputId = "num",
                  label = "Choose a number",
                  value = 25, min = 1,
                  max = 100)
    ),

    mainPanel(
      plotOutput("hist")
    )
  )
)

server <- function(input, output) {}

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



plotOutput("hist")

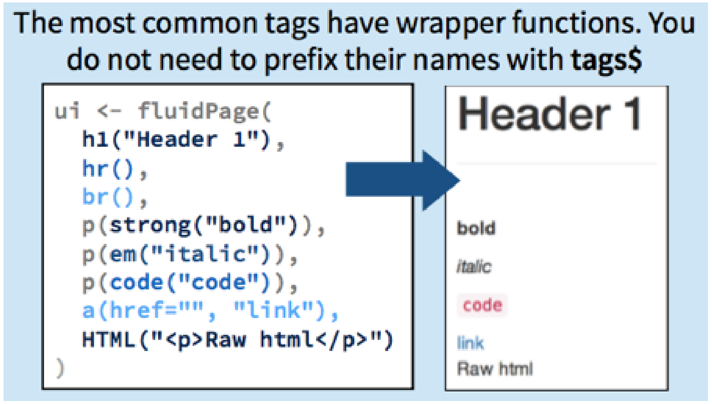
html tags if you know HTML



HTML

Add static HTML elements with **tags**, a list of functions that parallel common HTML tags, e.g. `tags$a()`. Unnamed arguments will be passed into the tag; named arguments will become tag attributes.

tags$a	tags$data	tags$h6	tags$nav	tags$span
tags$abbr	tags$datalist	tags$head	tags$noscript	tags$strong
tags$address	tags$dd	tags$header	tags$object	tags$style
tags$area	tags$del	tags$hgroup	tags$ol	tags$sub
tags$article	tags$details	tags$hr	tags$optgroup	tags$summary
tags$aside	tags$dfn	tags$HTML	tags$option	tags$sup
tags$audio	tags$div	tags$i	tags$output	tags$table
tags$b	tags$dl	tags$iframe	tags$p	tags$tbody
tags$base	tags$dt	tags$img	tags$param	tags$td
tags$bdi	tags$em	tags$input	tags$pre	tags$textarea
tags$bdo	tags$embed	tags$ins	tags$progress	tags$tfoot
tags$blockquote	tags$eventsource	tags$kbd	tags$q	tags$th
tags$body	tags$fieldset	tags$keygen	tags$ruby	tags$thead
tags$br	tags$figcaption	tags$label	tags$rp	tags$time
tags$button	tags$figure	tags$legend	tags$rt	tags$title
tags$canvas	tags$footer	tags$li	tags$s	tags$tr
tags$caption	tags$form	tags$link	tags$samp	tags$track
tags$cite	tags$h1	tags$mark	tags$script	tags$u
tags$code	tags$h2	tags$map	tags$section	tags$ul
tags$col	tags$h3	tags$menu	tags$select	tags$var
tags$colgroup	tags$h4	tags$meta	tags$small	tags$video
tags$command	tags$h5	tags$meter	tags$source	tags$wbr



The most common tags have wrapper functions. You do not need to prefix their names with **tags$**

```
ui <- fluidPage(
  h1("Header 1"),
  hr(),
  br(),
  p(strong("bold")),
  p(em("italic")),
  p(code("code")),
  a(href="", "link"),
  HTML("<p>Raw html</p>")
)
```

Header 1

bold

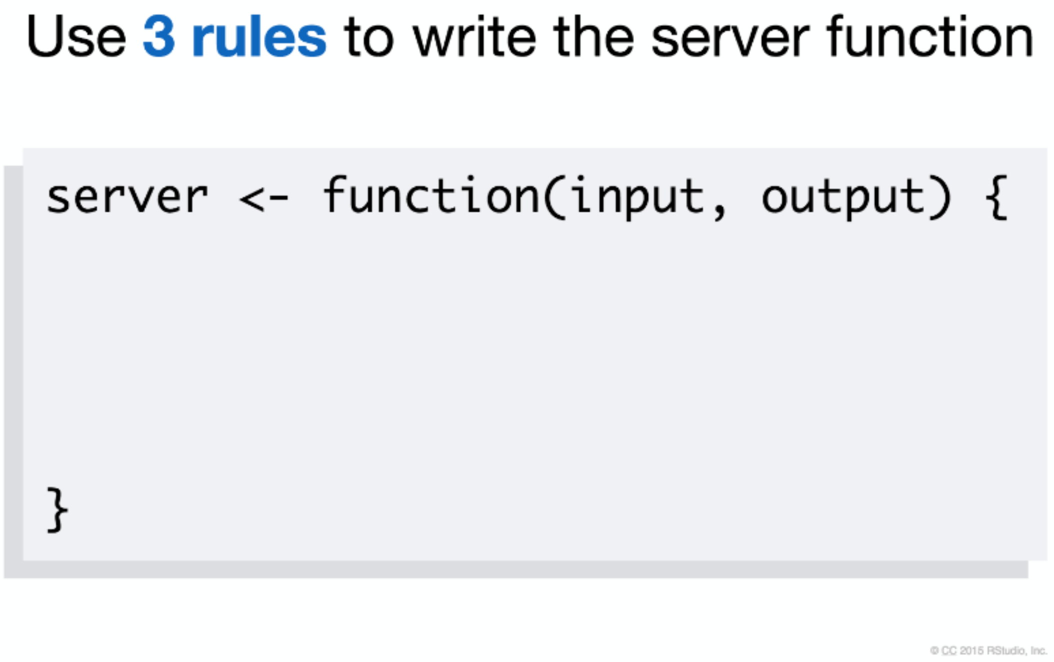
italic

`code`

link

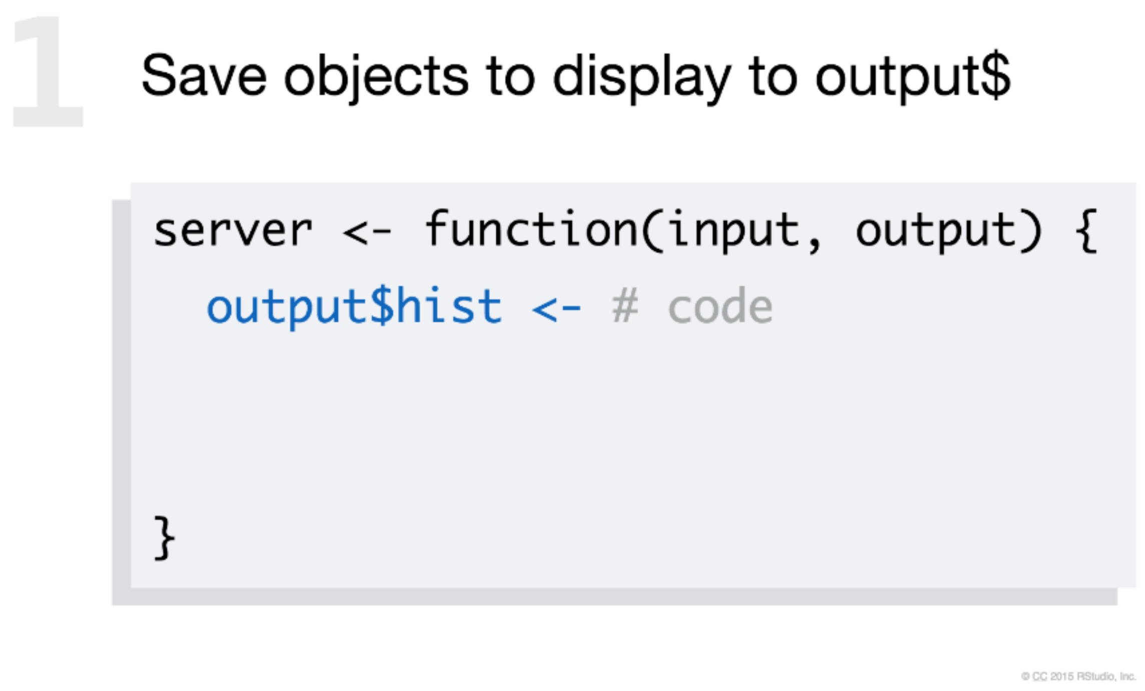
Raw html

Server: `server()` function



Use **3 rules** to write the server function

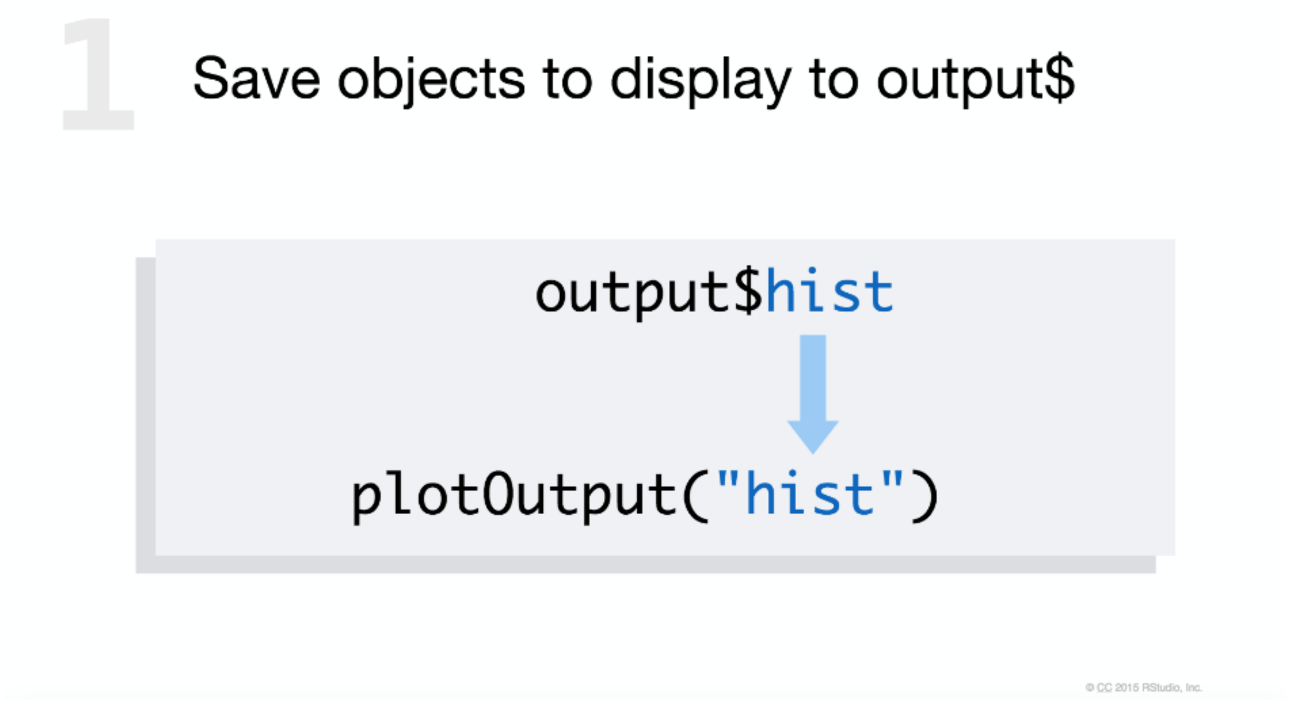
```r
server <- function(input, output) {



}
```



1

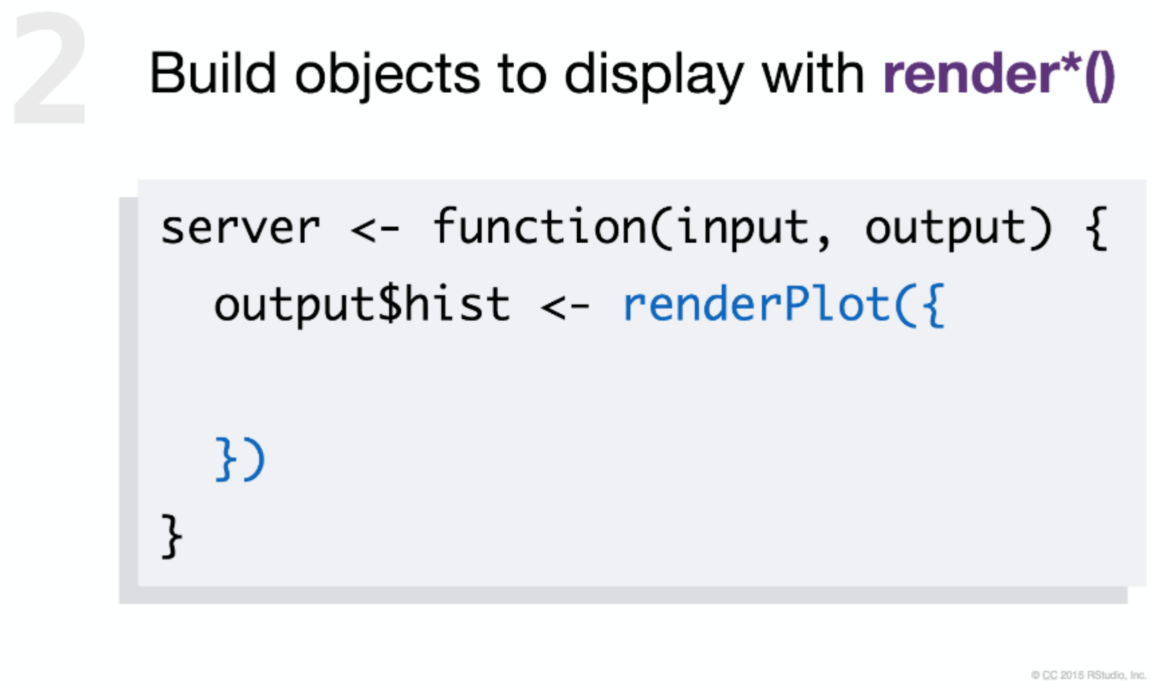
Save objects to display to output$

```r
server <- function(input, output) {
  output$hist <- # code



}
```



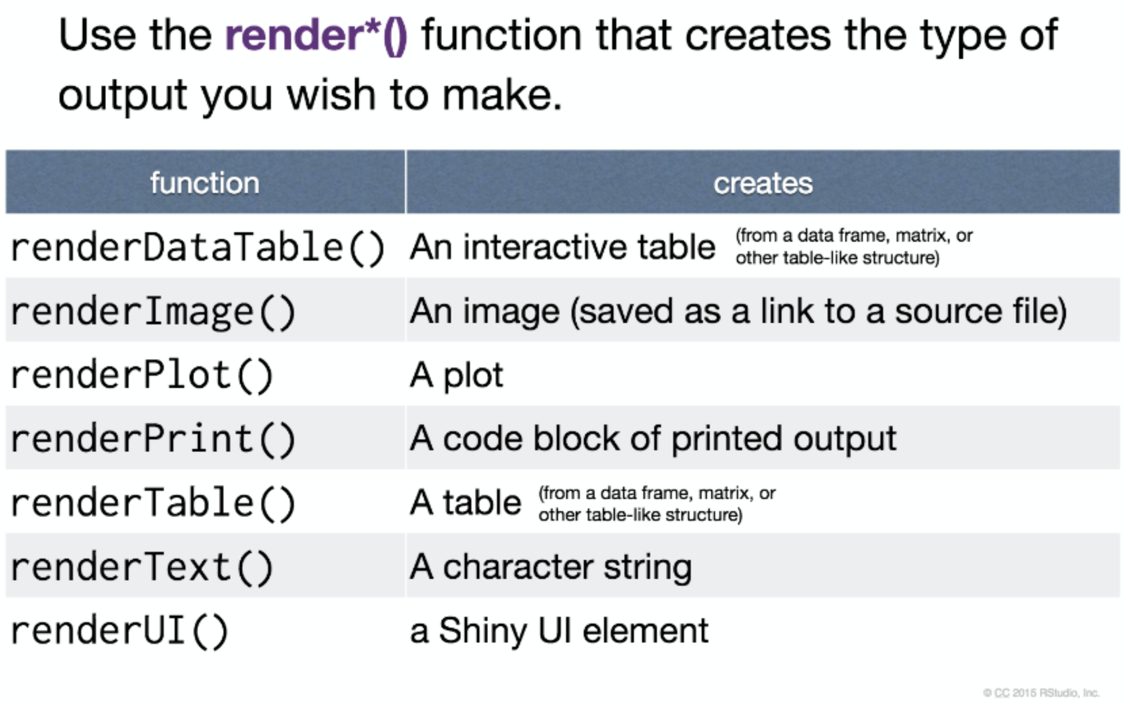
1 Save objects to display to output$

```
output$hist

plotOutput("hist")
```



2 Build objects to display with **render*()**

```r
server <- function(input, output) {
  output$hist <- renderPlot({

  })
}
```

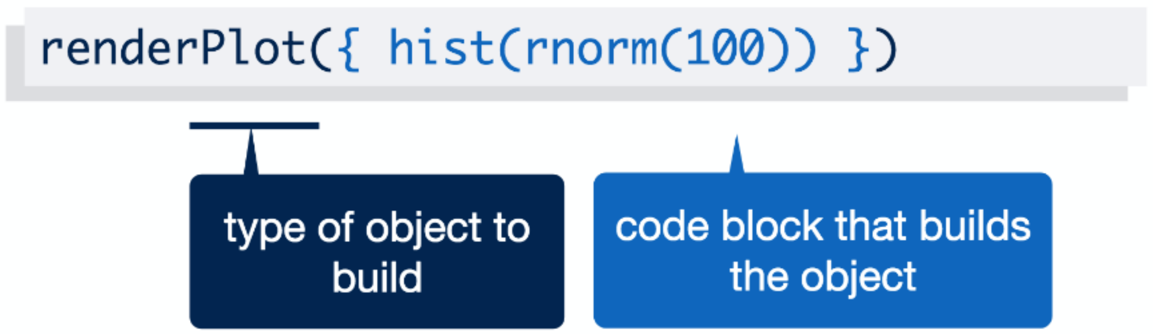


Use the **render*()** function that creates the type of output you wish to make.

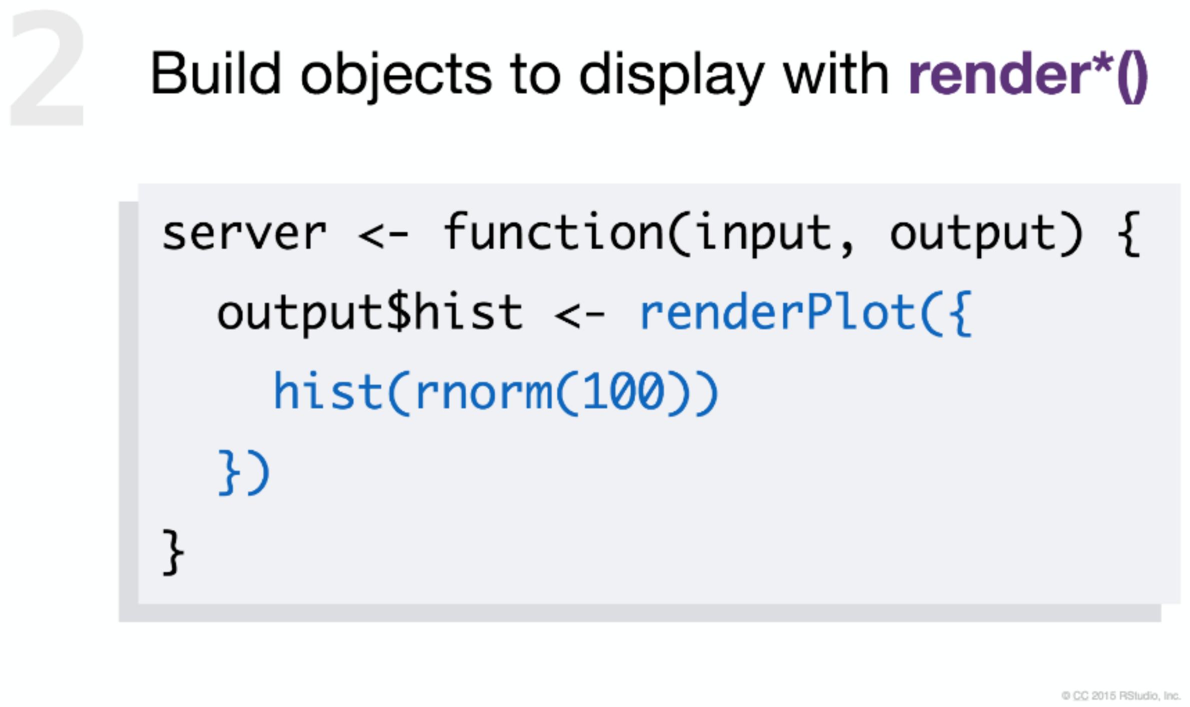
function	creates
`renderDataTable()`	An interactive table (from a data frame, matrix, or other table-like structure)
`renderImage()`	An image (saved as a link to a source file)
`renderPlot()`	A plot
`renderPrint()`	A code block of printed output
`renderTable()`	A table (from a data frame, matrix, or other table-like structure)
`renderText()`	A character string
`renderUI()`	a Shiny UI element

render*()

Builds reactive output to display in UI

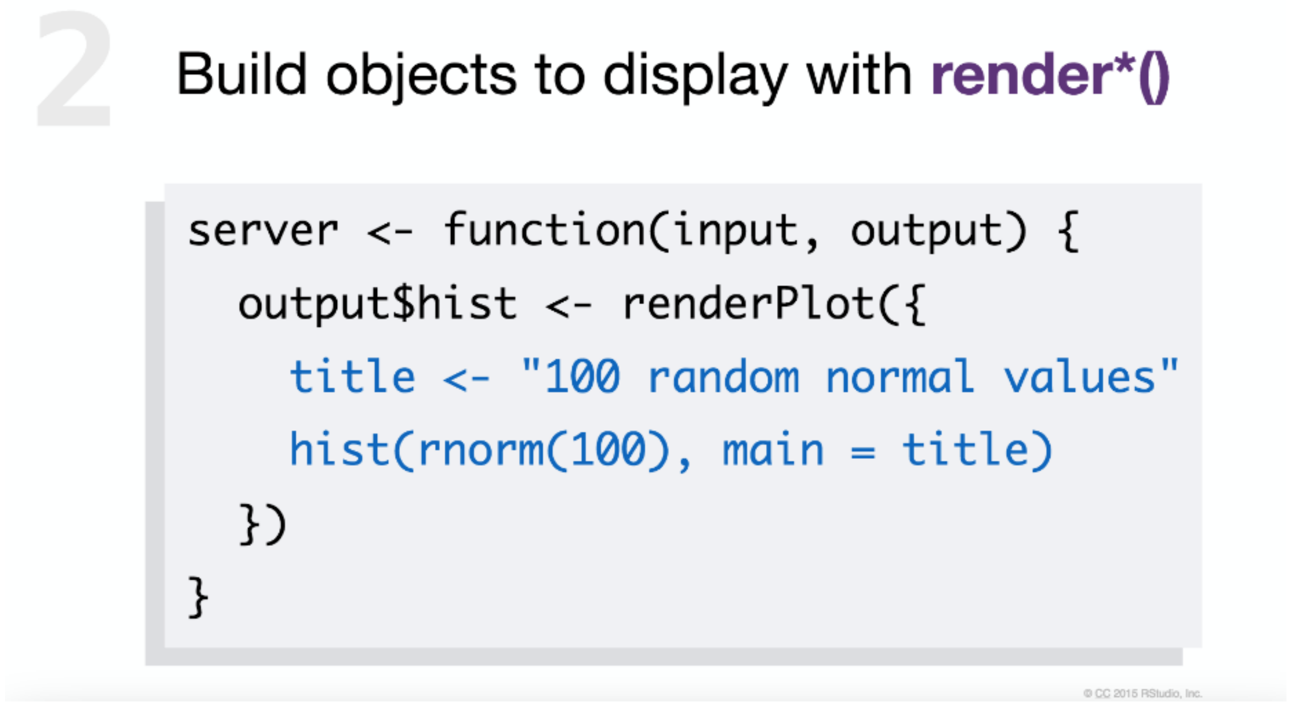


```r
renderPlot({ hist(rnorm(100)) })
```



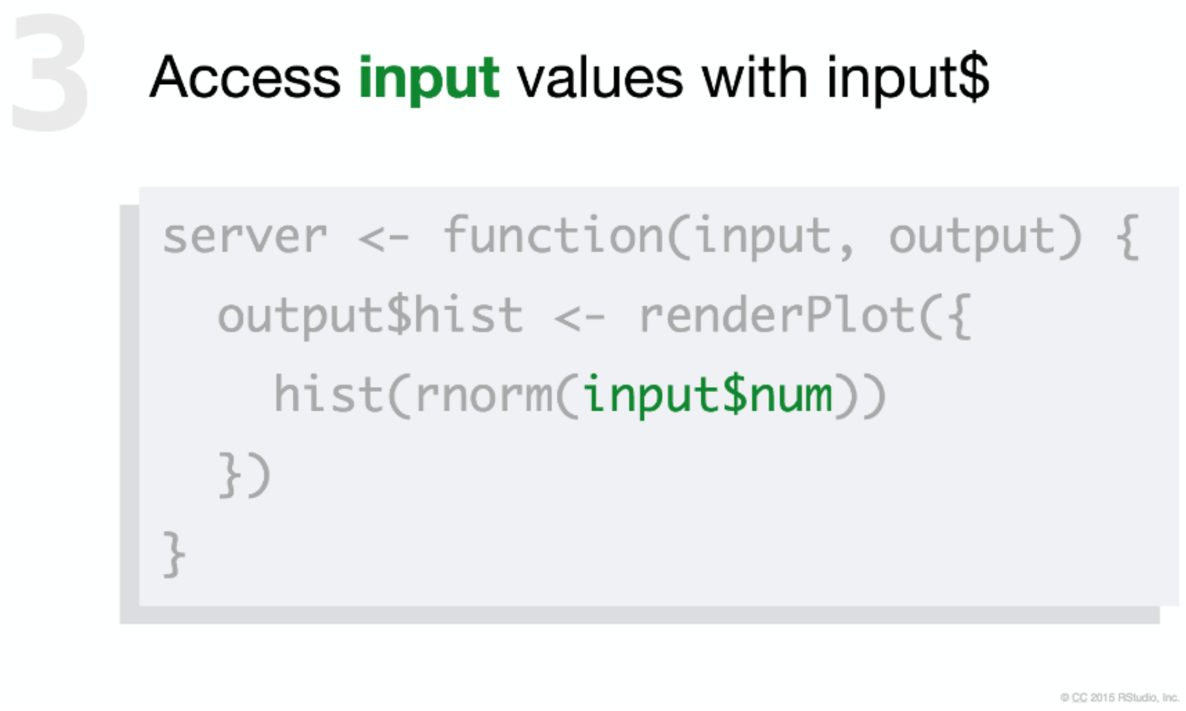
2 Build objects to display with **render*()**

```r
server <- function(input, output) {
  output$hist <- renderPlot({
    hist(rnorm(100))
  })
}
```



2 Build objects to display with **render*()**

```
server <- function(input, output) {
  output$hist <- renderPlot({
    title <- "100 random normal values"
    hist(rnorm(100), main = title)
  })
}
```



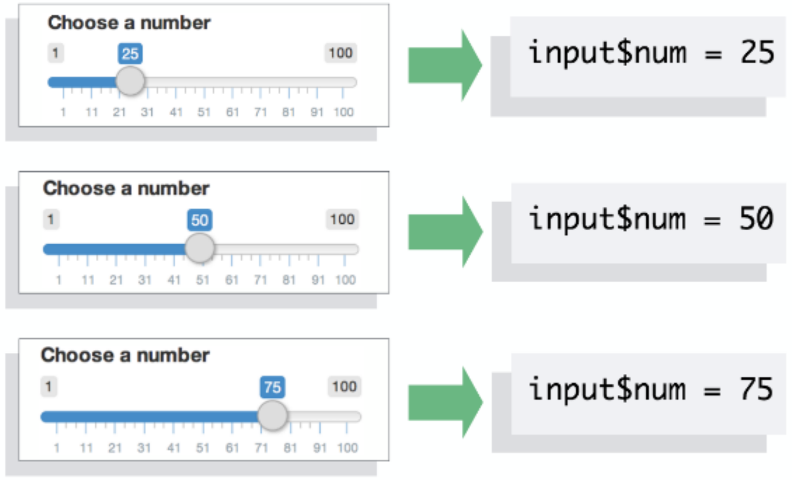
3

Access **input** values with input$

```r
server <- function(input, output) {
  output$hist <- renderPlot({
    hist(rnorm(input$num))
  })
}
```

Input values

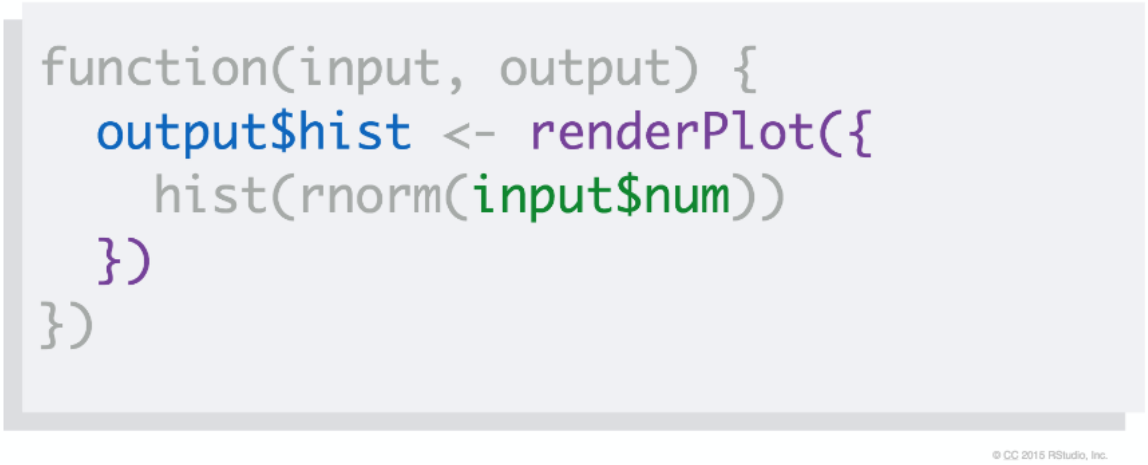
The input value changes whenever a user changes the input.



```
input$num = 25
input$num = 50
input$num = 75
```

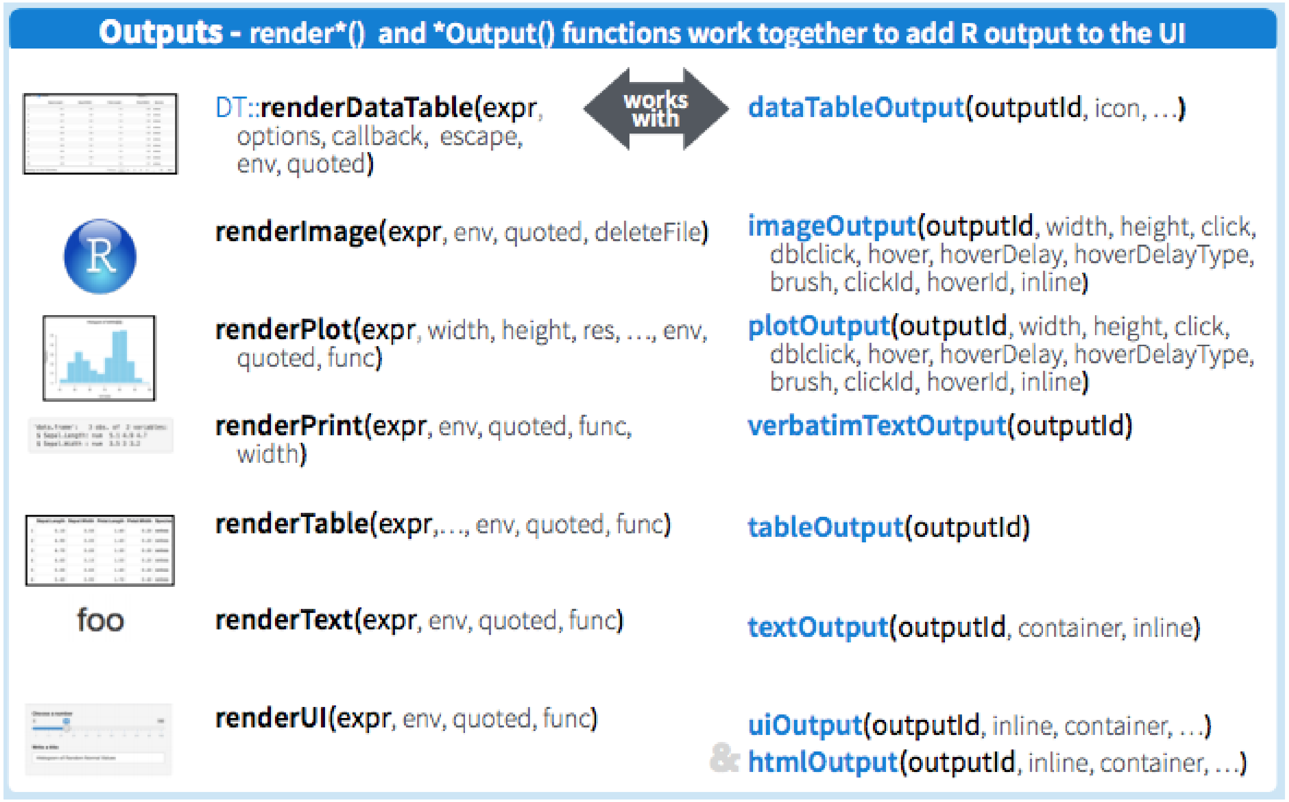
Reactivity 101

Reactivity automatically occurs whenever you use an input value to render an output object



```r
function(input, output) {
  output$hist <- renderPlot({
    hist(rnorm(input$num))
  })
})
```

Shiny: Output

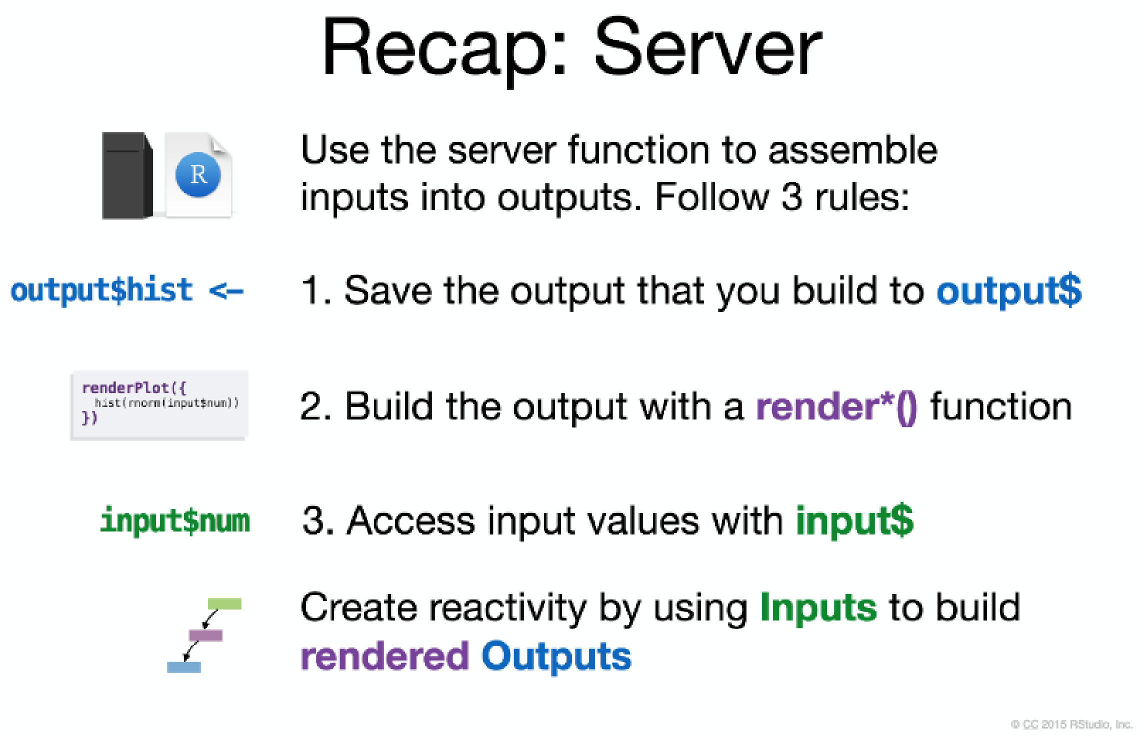


Outputs - render*() and *Output() functions work together to add R output to the UI

works with

	DT::**renderDataTable(expr**, options, callback, escape, env, quoted**)**	**dataTableOutput(outputId**, icon, …**)**
	renderImage(expr, env, quoted, deleteFile**)**	**imageOutput(outputId**, width, height, click, dblclick, hover, hoverDelay, hoverDelayType, brush, clickId, hoverId, inline**)**
	renderPlot(expr, width, height, res, …, env, quoted, func**)**	**plotOutput(outputId**, width, height, click, dblclick, hover, hoverDelay, hoverDelayType, brush, clickId, hoverId, inline**)**
	renderPrint(expr, env, quoted, func, width**)**	**verbatimTextOutput(outputId)**
	renderTable(expr,…, env, quoted, func**)**	**tableOutput(outputId)**
foo	**renderText(expr**, env, quoted, func**)**	**textOutput(outputId**, container, inline**)**
	renderUI(expr, env, quoted, func**)**	**uiOutput(outputId**, inline, container, …**)** & **htmlOutput(outputId**, inline, container, …**)**

Server Recap



Recap: Server

Use the server function to assemble inputs into outputs. Follow 3 rules:

`output$hist <-` 1. Save the output that you build to **output$**

```
renderPlot({
  hist(rnorm(input$num))
})
```

2. Build the output with a **render*()** function

`input$num` 3. Access input values with **input$**

Create reactivity by using **Inputs** to build **rendered Outputs**

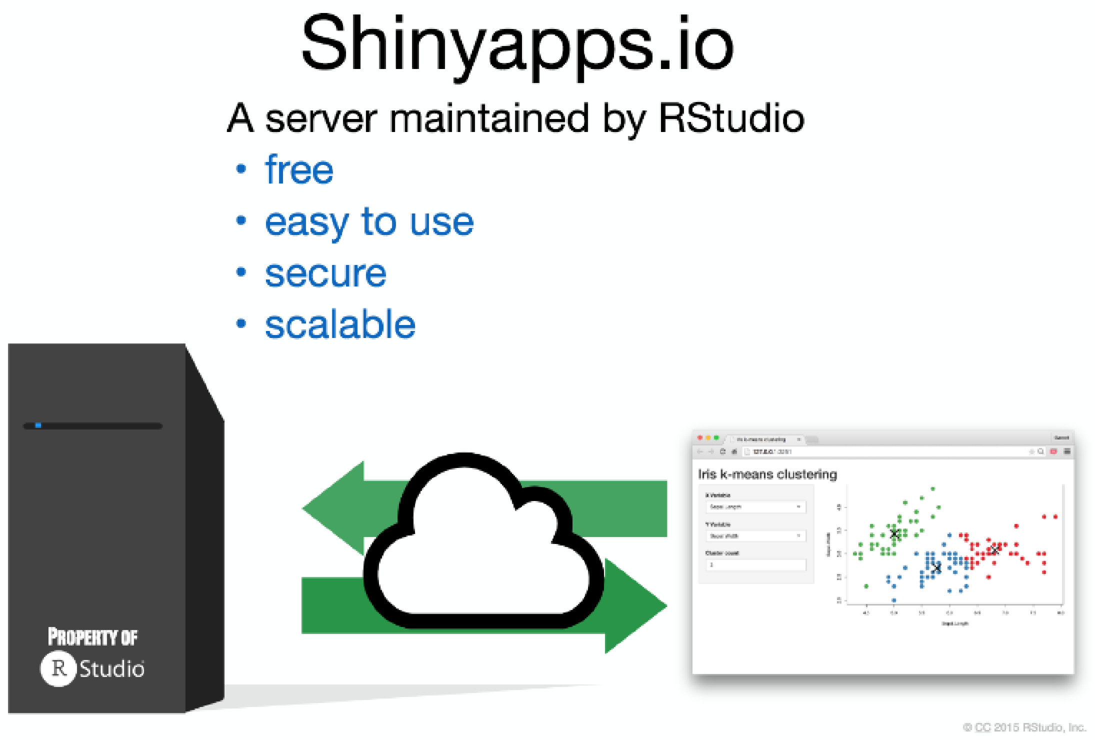
15 minute Quick Assignment

Open the `app.R` file (click here).

Try these three tasks:

1. add a new slider that sets the number of breaks for the rnorm() function
2. add a textInput() that sets the name of the plot
3. add a actionButton() that updates the name of the plot (part 2) only when clicking (hint: see ?observeEvent)

Deploying Apps to Shinyapps.io

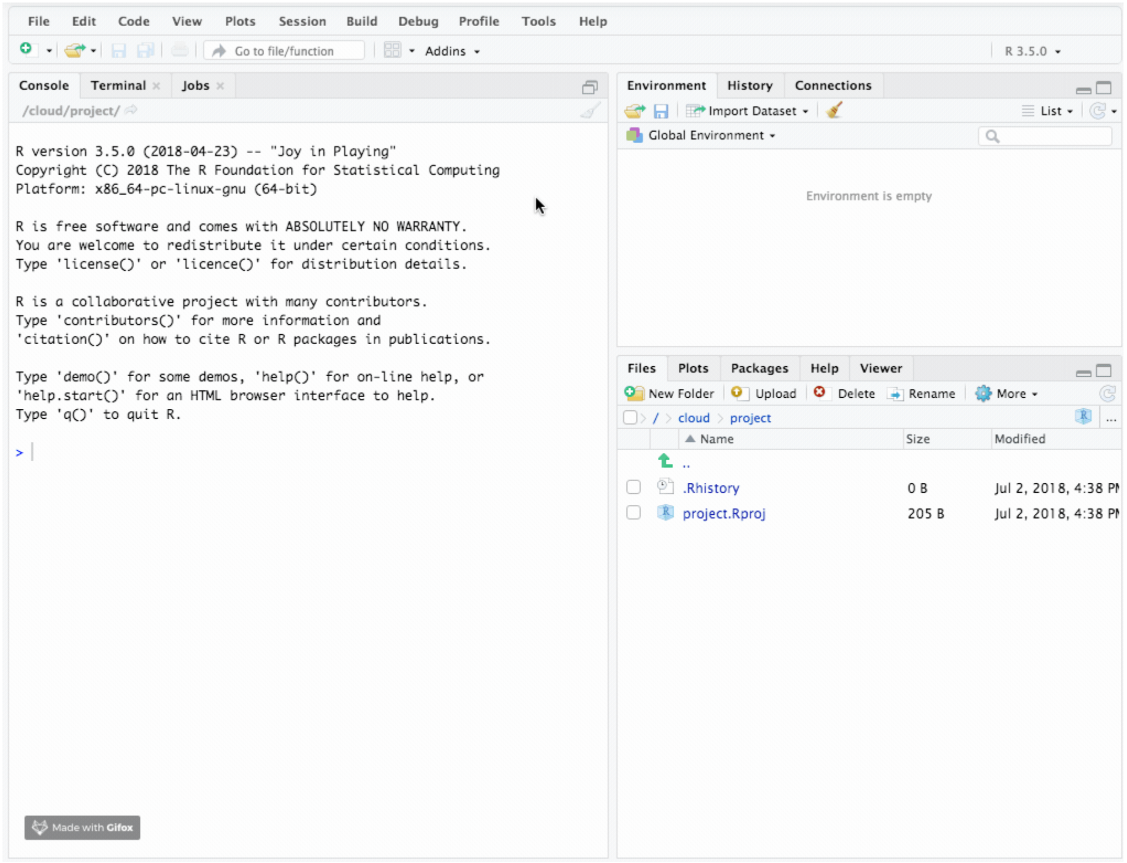


Shinyapps.io

A server maintained by RStudio

- free
- easy to use
- secure
- scalable

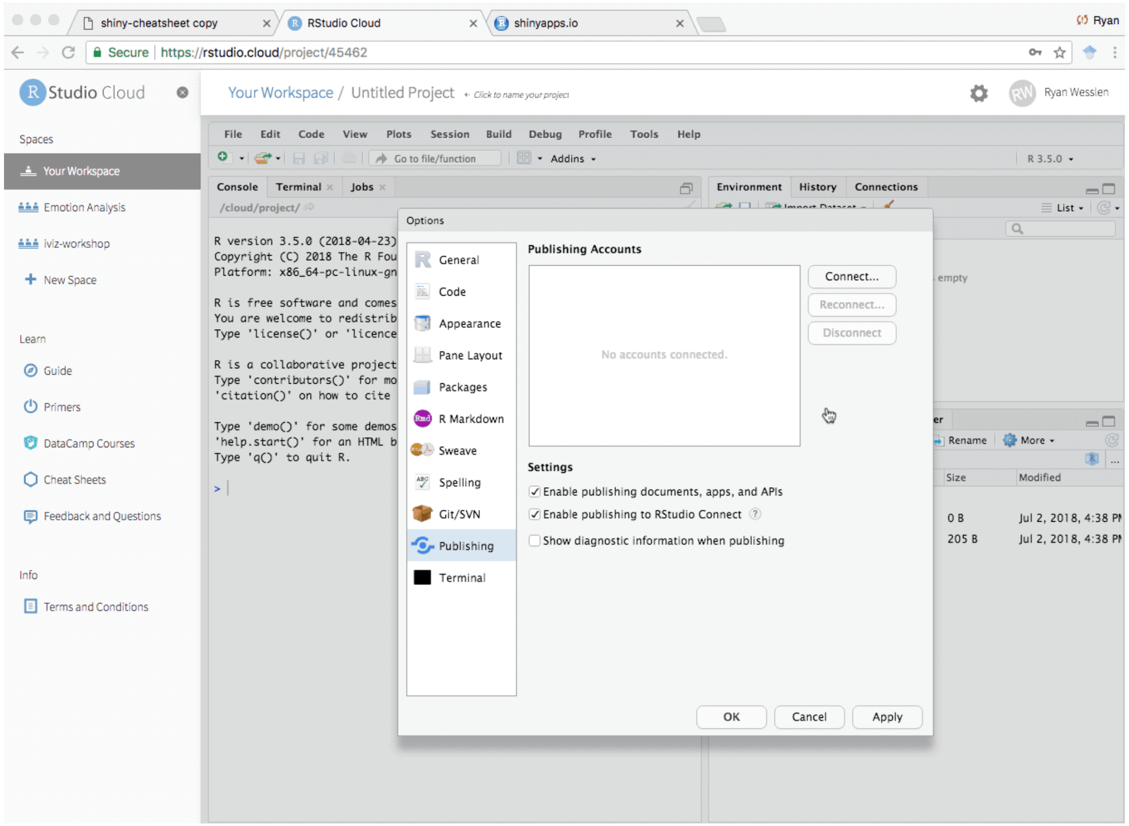
Setup connection to ShinyApps.io



```
R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

Name	Size	Modified
..		
.Rhistory	0 B	Jul 2, 2018, 4:38 PM
project.Rproj	205 B	Jul 2, 2018, 4:38 PM

Setup connection to ShinyApps.io



```
R version 3.5.0 (2018-04-23)
Copyright (C) 2018 The R Fou
Platform: x86_64-pc-linux-gn

R is free software and comes
You are welcome to redistrib
Type 'license()' or 'licence

R is a collaborative project
Type 'contributors()' for mo
'citation()' on how to cite

Type 'demo()' for some demos
'help.start()' for an HTML b
Type 'q()' to quit R.

>
```

Publishing Accounts

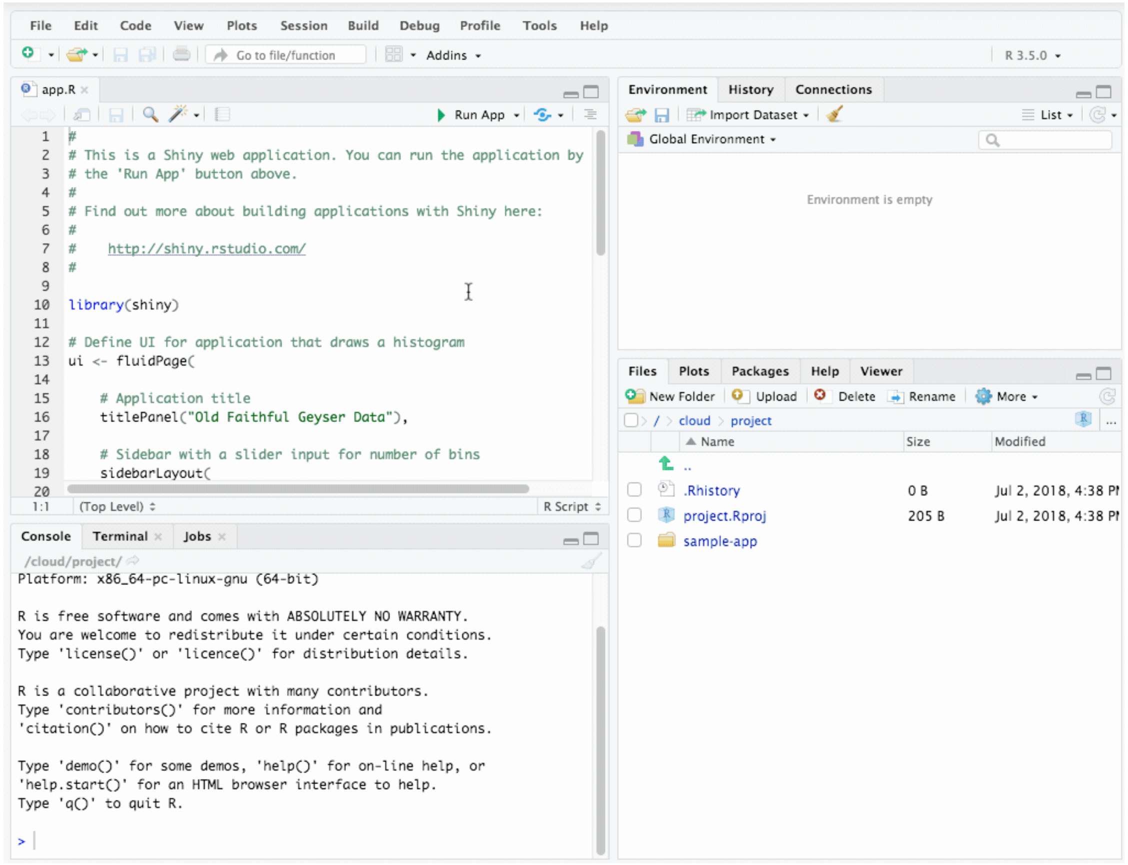
No accounts connected.

Settings

- ✓ Enable publishing documents, apps, and APIs
- ✓ Enable publishing to RStudio Connect
- ☐ Show diagnostic information when publishing

Size	Modified
0 B	Jul 2, 2018, 4:38 PM
205 B	Jul 2, 2018, 4:38 PM

Deploy to ShinyApps.io

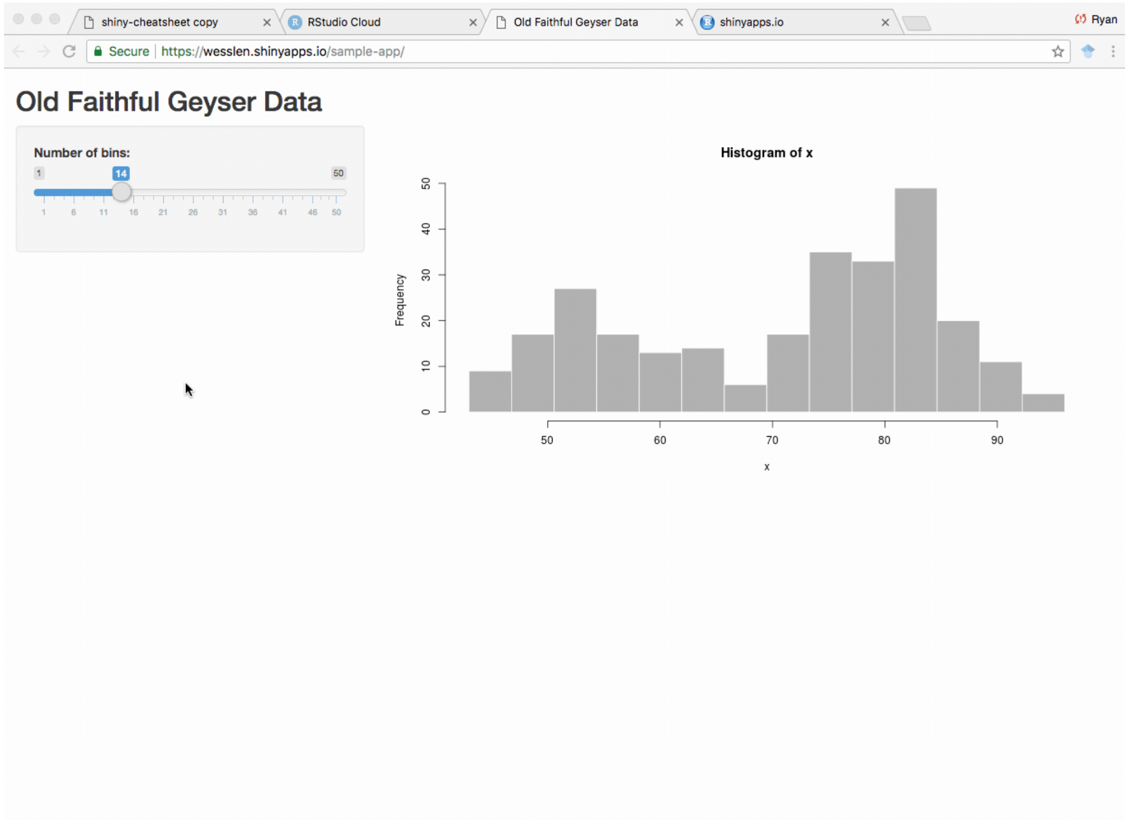


```r
#
# This is a Shiny web application. You can run the application by
# the 'Run App' button above.
#
# Find out more about building applications with Shiny here:
#
#    http://shiny.rstudio.com/
#

library(shiny)

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

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

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

Name	Size	Modified
.Rhistory	0 B	Jul 2, 2018, 4:38 PM
project.Rproj	205 B	Jul 2, 2018, 4:38 PM
sample-app		

```
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

Deploy to ShinyApps.io


shiny-cheatsheet copy
RStudio Cloud
Old Faithful Geyser Data
shinyapps.io
Ryan
Secure | https://wesslen.shinyapps.io/sample-app/
Old Faithful Geyser Data
Number of bins:
1
14
50
Histogram of x
Frequency
x