



MANGO  
SOLUTIONS

# Introduction to Shiny

LondonR – Workshop June 27<sup>th</sup> 2018

Nicolas Attalides

Data Scientist

✉ [nattalides@mango-solutions.com](mailto:nattalides@mango-solutions.com)



# WiFi

Network Name: UCLGuest

or use: [guest.ucl.ac.uk/portal](https://guest.ucl.ac.uk/portal)

Go to self-service and fill in the form.

Your email address is your username and password is generated once you submit.

Event code: **“londonr”**



# Workshop structure

- 2 hours
- Presentation format
- Worked examples of creating apps
- Exercises during the workshop



# Workshop resources

- R (version 3.1.2)
- RStudio
- Shiny (version 0.11)



# Workshop Aim

Be able to develop a simple Shiny App with standard inputs and outputs.



# Outline

- A Basic Shiny app
- The User Interface script
- The Server script
- Inputs & Outputs
- Reactivity
- Beyond the Basics
- Shiny Themes



# What is Shiny?

- R Package for Interactive Web Apps developed by RStudio
- Gives the power of R in a convenient user interface
- Can be written entirely in R without the need of web development skills



# A Basic Shiny App

- A basic app requires:
  - A User Interface script – `ui.R`
  - A Server script – `server.R`
- Runs using the `runApp` function





# The User Interface Script

- Defines the components of the user interface:
  - Page titles
  - Inputs
  - Outputs
- Contains what the user will see and interact with
- Requires a user interface object



# The Server Script

- Defines the server-side logic of the application and what happens in R
- Contains the information to build the app in the form of functions that map user inputs to outputs
- Requires a function with arguments `input` and `output`



# Worked Example 1

## My First Shiny App!

**Enter text here:**

You entered the text: Welcome to LondonR!



# Worked Example 1 – ui.R

```
fluidPage(  
  titlePanel("My First Shiny App!"),  
  sidebarLayout(  
    sidebarPanel(  
      textInput("myText", "Enter text here:")  
    ),  
    mainPanel(  
      textOutput("niceTextOutput")  
    )  
  )  
)
```

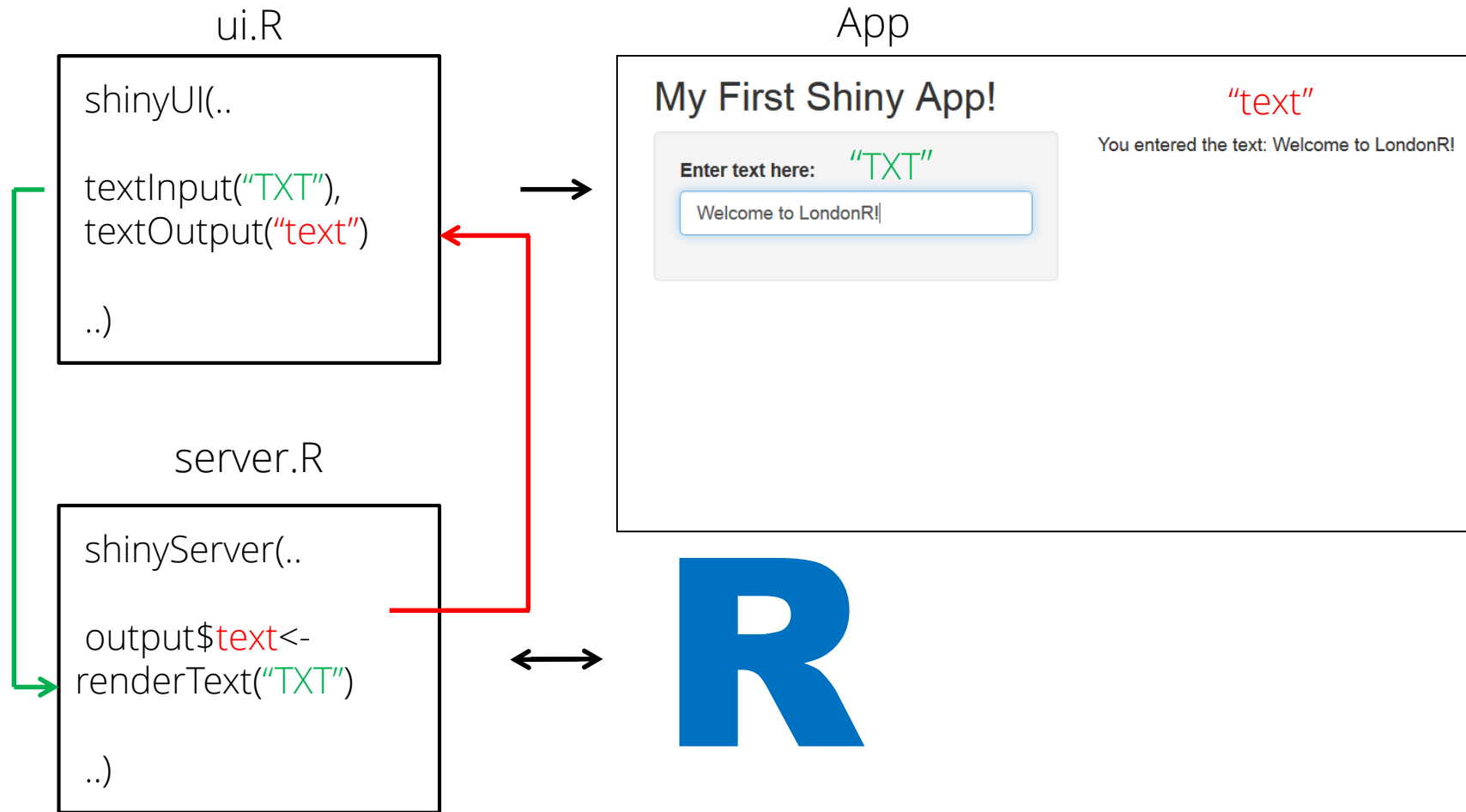


# Worked Example 1 – server.R

```
function(input, output){  
  output$niceTextOutput <- renderText({  
    paste("You entered the text:", input$myText)  
  })  
}
```



# Schematic of a Basic Shiny app



# Layouts

- Example 1 uses `sidebarLayout()`
- There are a number of possible layouts
- In this workshop we will only use
  - `sidebarPanel()`
    - Useful for `..Input()` functions
  - `mainPanel()`
    - Useful for `..Output()` functions



# ui.R - Boiler Plate

```
fluidPage(  
  titlePanel("Title Here!"),  
  sidebarLayout(  
    sidebarPanel(  
      #INPUTS GO HERE  
    ),  
    mainPanel(  
      #OUTPUTS GO HERE  
    )  
  )  
)
```





# server.R - Boiler Plate

```
function(input, output){  
  #CODE GO HERE  
}
```



# Input Controls

Input	Description
<code>textInput()</code>	Text string input
<code>numericInput()</code>	Numeric value input
<code>selectInput()</code>	Select single or multiple values from drop down list
<code>sliderInput()</code>	Numeric (single or range) "slider" input
<code>radioButtons()</code>	Set of radio button inputs
<code>fileInput()</code>	File upload control



# Worked Example 2

## My First Shiny App!

**Enter text here:**

**Select a number:**



**Select from the dropdown:**



# Worked Example 2 – ui.R

```
sidebarPanel(  
  textInput("myTextInput", "Enter text here:"),  
  
  numericInput("myNumberInput", "Select a number:",  
    value = 50, min = 0, max = 100, step = 1),  
  
  selectInput("mySelectInput", "Select from the dropdown:",  
    choices = LETTERS[1:10])  
)
```



# HTML Formatting

- We don't need to use HTML tags
- Shiny includes a series of equivalent functions

Function	Usage
<code>p()</code>	A paragraph of text
<code>h*()</code>	A level * (1, 2, 3,...) header
<code>code()</code>	A block of code
<code>img()</code>	An image
<code>strong()</code>	Bold text
<code>em()</code>	Italic text



# Worked Example 2

## My First Shiny App!

Enter text here:

Select a number:

Select from the dropdown:

### Using HTML in Shiny

This is a paragraph of text that is included in our main panel. **This text will be in bold.**

You entered the text: Welcome to LondonR!

You selected the number: 50

You selected option: A



# Worked Example 2 – ui.R

```
mainPanel(  
  h4("Using HTML in Shiny"),  
  
  p("This is a paragraph of text that is included in our  
    main panel.", strong("This text will be in bold.")),  
  
  textOutput("niceTextOutput"),  
  
  textOutput("niceNumberOutput"),  
  
  textOutput("niceSelectOutput")  
)
```



# Worked Example 2 – ui.R

```
fluidPage(  
  titlePanel("My First Shiny App!"),  
  sidebarLayout(  
    sidebarPanel(  
      textInput("myTextInput", "Enter text here:"),  
      numericInput("myNumberInput", "Select a number:",  
                   value = 50, min = 0, max = 100, step = 1),  
      selectInput("mySelectInput", "Select from the dropdown:",  
                  choices = LETTERS[1:10])  
    ),  
    mainPanel(  
      h4("Using HTML in Shiny"),  
      p("This is a paragraph of text that is included in our main panel.",  
        strong("This text will be in bold.")),  
      textOutput("niceTextOutput"),  
      textOutput("niceNumberOutput"),  
      textOutput("niceSelectOutput")  
    )  
  )  
)
```





# Worked Example 2 – server.R

```
function(input, output){  
  output$niceTextOutput <- renderText({  
    paste("You entered text: ", input$myTextInput)  
  })  
  
  output$niceNumberOutput <- renderText({  
    paste("You selected the number: ", input$myNumberInput)  
  })  
  
  output$niceSelectOutput <- renderText({  
    paste("You selected option:", input$mySelectInput)  
  })  
}
```



# Exercise 1

Build a simple Shiny application that takes a date input and returns the following text:

- What day of the week is it (e.g. “Wednesday”)
- What month it is (e.g. “December”)
- What year it is

Hint: `format(Sys.Date(), "Day: %A Month: %B Year: %Y")`



# Exercise 1 – ui.R

```
library(shiny)
```

```
fluidPage(  
  titlePanel("Exercise 1"), # Define the header for the page  
  sidebarLayout( # Set up the page to have a sidebar  
    sidebarPanel(  
      # Define the contents of the sidebar  
      dateInput("dateInput", "Select a date:")  
    ),  
    mainPanel(  
      # Define the contents of the main panel  
      textOutput("dateOutput")  
    )  
  )  
)
```



# Exercise 1 – server.R

```
function(input, output){  
  output$dateOutput <- renderText({  
    format(input$dateInput,  
           format = "A %A in %B. The year is %Y")  
  })  
}
```



# Defining Outputs

- So far we have just output text
- Shiny also allows us to output graphics, data and images
- We have to define the output in the UI and the Server scripts using different functions



# Rendering Outputs

Output Type	server.R Function	ui.R Function
Text	<code>renderText()</code>	<code>textOutput()</code>
Data	<code>renderDataTable()</code>	<code>dataTableOutput()</code>
Plot	<code>renderPlot()</code>	<code>plotOutput()</code>
Image	<code>renderImage()</code>	<code>imageOutput()</code>



# Worked Example 3 - Render Data

- From the user interface select a dataset from a dropdown menu using `selectInput` in `ui.R`
- Render the data using `renderDataTable` in `server.R`
- Display the data in a table using `dataTableOutput` in `ui.R`



# Worked Example 3 - Render Data

## Render Data in a Shiny App

Select from the dropdown:

airquality

Show 25 entries Search:

Ozone	Solar.R	Wind	Temp	Month	Day
41	190	7.4	67	5	1
36	118	8	72	5	2
12	149	12.6	74	5	3
18	313	11.5	62	5	4
		14.3	56	5	5
28		14.9	66	5	6
23	299	8.6	65	5	7
19	99	13.8	59	5	8
8	19	20.1	61	5	9
	194	8.6	69	5	10
7		6.9	74	5	11
16	256	9.7	69	5	12
11	290	9.2	66	5	13
14	274	10.9	68	5	14





# Worked Example 3 – ui.R

```
fluidPage(  
  titlePanel("My First Shiny App!"),  
  sidebarLayout(  
    sidebarPanel(  
      selectInput("selectInput",  
                  "Select from the dropdown:",  
                  choices = c("airquality", "iris", "mtcars"))  
    ),  
    mainPanel(  
      dataTableOutput("dataOutput")  
    )  
  )  
)
```



# Worked Example 3 – server.R

```
function(input, output){  
  output$dataOutput <- renderDataTable({  
    switch(input$selectInput,  
      "airquality" = airquality,  
      "iris" = iris,  
      "mtcars" = mtcars)  
  })  
}
```



# Worked Example 4 - Render Plots

- From the user interface select a column of the `mtcars` dataset from a drop down menu using `selectInput` in `ui.R`
- Plot a histogram of the data

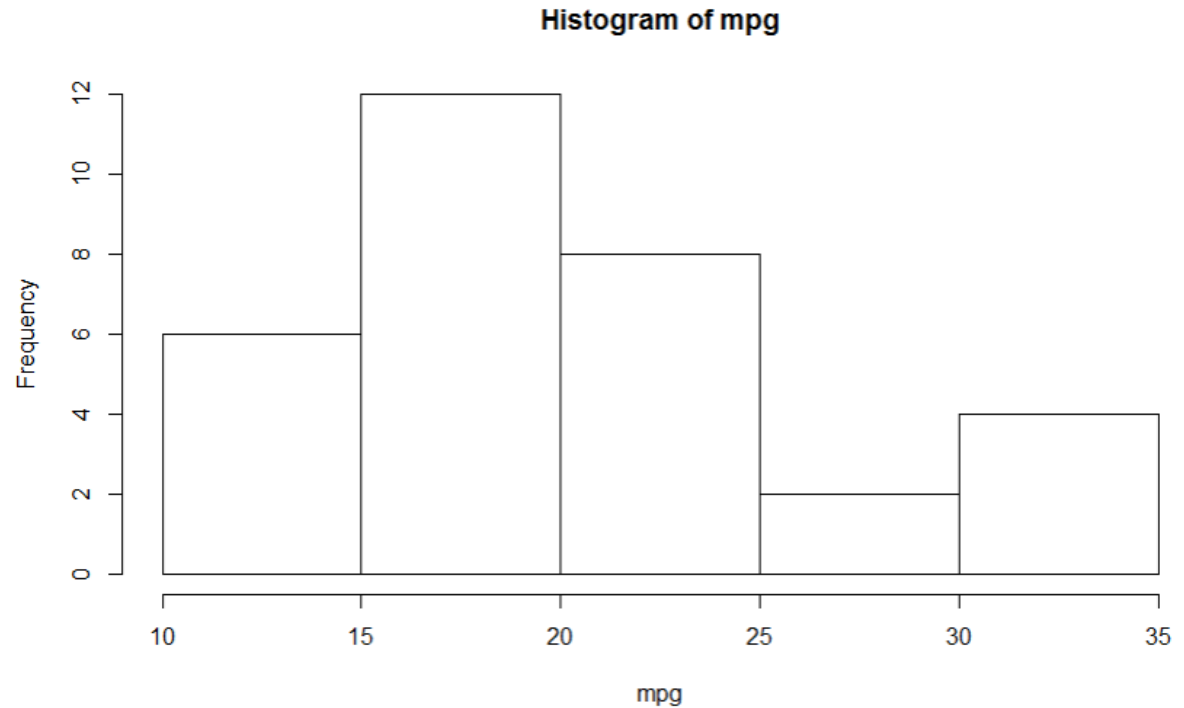


# Worked Example 4 - Render Plots

## Render Plot in a Shiny App

Select column:

mpg ▼



# Worked Example 4 – ui.R

```
fluidPage(  
  titlePanel("My First Shiny App!"),  
  sidebarLayout(  
    sidebarPanel(  
      selectInput("selectInput", "select column:",  
                  choices = colnames(mtcars))  
    ),  
    mainPanel(  
      plotOutput("plotOutput")  
    )  
  )  
)
```



# Worked Example 4 – server.R

```
function(input, output){  
  output$plotOutput <- renderPlot({  
    hist(mtcars[,input$selectInput],  
        main = paste("Histogram of", input$selectInput),  
        xlab = input$selectInput)  
  })  
}
```



# Exercise 2

Create a Shiny application that takes:

- A numeric input between 1 and 500
- A dropdown list input containing colours “red”, “yellow” and “blue”

Use these inputs to create an output histogram of randomly generated data from any distribution (e.g. normal) where  $n$  is the numeric input and histogram colour is the one chosen by the user.



# Exercise 2 – ui.R

```
fluidPage(  
  titlePanel("Exercise 2 - Render Plot in a Shiny App"),  
  sidebarLayout(  
    sidebarPanel(  
      numericInput("numberInput", "select size of data:",  
                   min = 1, max = 500, value = 100),  
      selectInput("colInput", "select a colour:",  
                  choices = c("red", "yellow", "blue"))  
    ),  
    mainPanel(  
      plotOutput("plotOutput")  
    )  
  )  
)
```





# Exercise 2 – server.R

```
function(input, output){  
  output$plotOutput <- renderPlot({  
    hist(rnorm(input$numberInput),  
         col = input$colInput)  
  })  
}
```



# Reactivity - Question

Consider Exercise 2...

- Suppose we want to change the colour of the plot (from red to blue), what happens to the data?



# Reactivity

- Each time we change an option (in the UI) the data is simulated again
- Suppose this was reading in a large dataset, connecting to a database etc.



# The `reactive()` function

- Limit the re-running of code using reactive expressions
- The `reactive()` function allows us to create a reactive expression
- The function is only called when the relevant inputs are updated



# Worked Example 5

## Render Plot in a Shiny App

Select size of data:

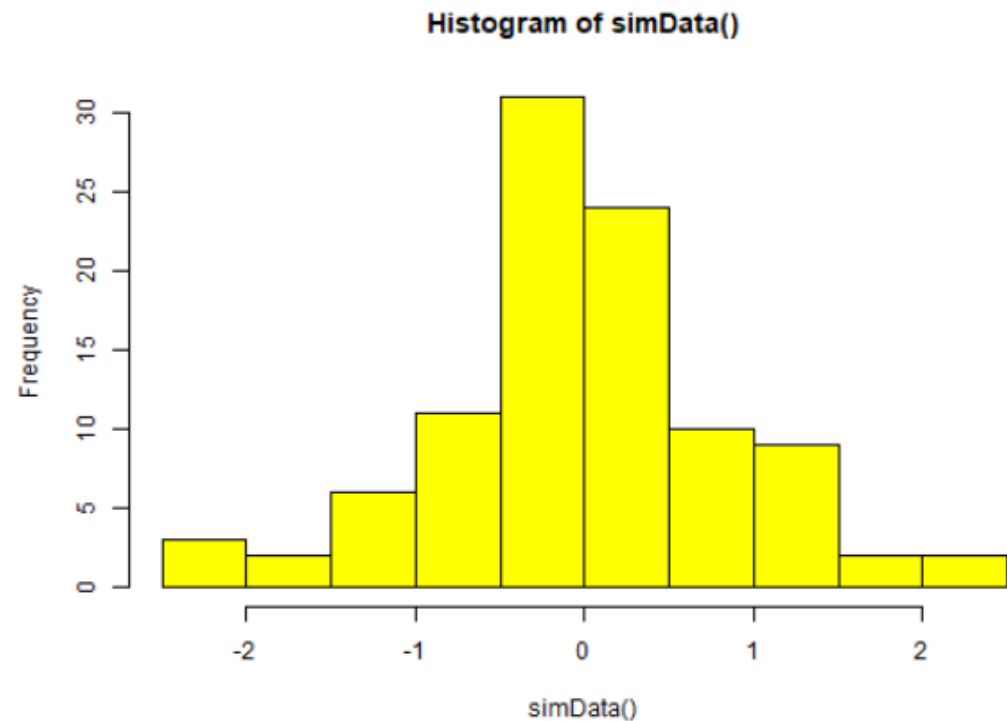
Select a colour:

yellow

red

yellow

blue



# Worked Example 5 – ui.R

```
fluidPage(  
  titlePanel("Render Plot in a Shiny App"),  
  sidebarLayout(  
    sidebarPanel(  
      numericInput("numberInput", "select size of data:",  
                   min = 1, max = 500, value = 100),  
      selectInput("colInput", "select a colour:",  
                  choices = c("red", "yellow", "blue"))  
    ),  
    mainPanel(  
      plotOutput("plotOutput")  
    )  
  )  
)
```



# Worked Example 5 – server.R

```
function(input, output){  
  simData <- reactive({  
    rnorm(input$numberInput)  
  })  
  
  output$plotOutput <- renderPlot({  
    hist(simData(), col = input$colInput)  
  })  
}
```

- Our data is only updated when the number of simulations is changed



# Beyond the basics

- Changes to layouts
- Including tabbed pages
- Include CSS to style the page
- Incorporate Shiny and Markdown
- Share your app
- ...

All covered on our 1 day **Introduction to Shiny** course





# Shiny Themes

- A new package that allows us to change the bootstrap theme
- Requires Shiny v0.11
- Available on CRAN

<http://rstudio.github.io/shinythemes/>



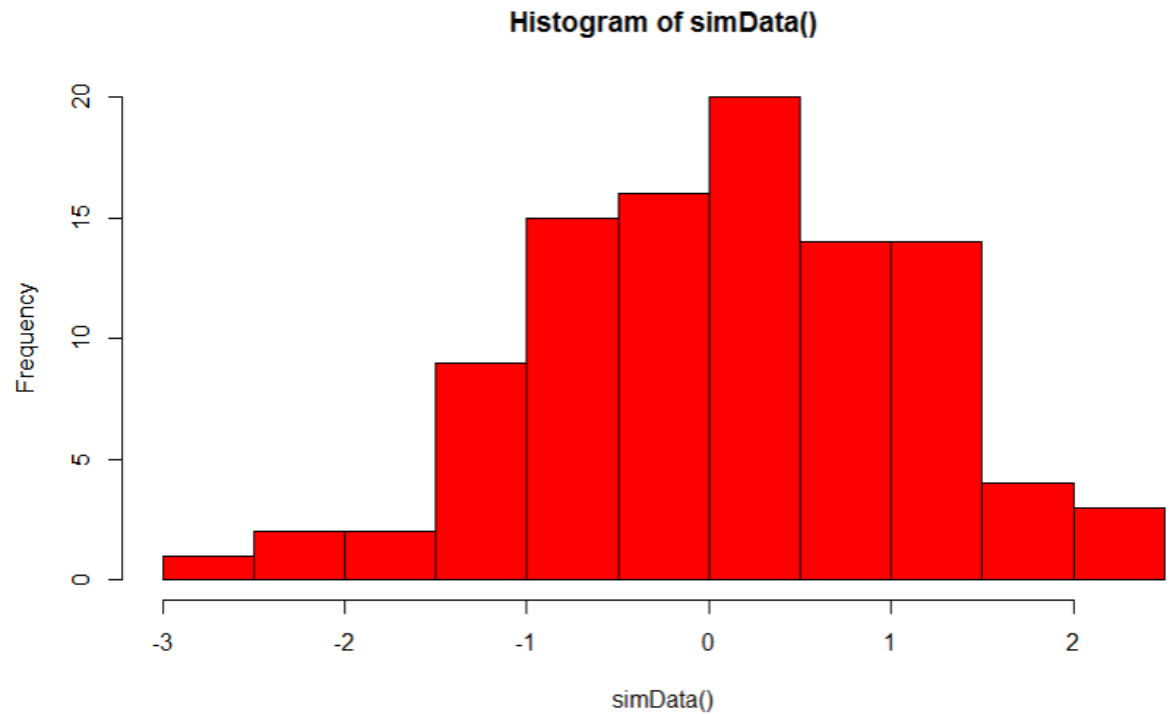
# Worked Example 6

## Render Plot in a Shiny App

Select size of data:

Select a colour:



# Worked Example 6 – ui.R

```
library(shinythemes)

fluidPage(
  theme = shinytheme("cerulean"),
  titlePanel("Render Plot in a Shiny App"),
  sidebarLayout(
    sidebarPanel(
      numericInput("numberInput", "Select size of data:",
                  min = 1, max = 500, value = 100),
      selectInput("colInput", "Select a colour:",
                 choices = c("red", "yellow", "blue"))
    ),
    mainPanel(
      plotOutput("plotOutput")
    )
  )
)
```



# Worked Example 6 – server.R

```
function(input, output){  
  simData <- reactive({  
    rnorm(input$numberInput)  
  })  
  
  output$plotOutput <- renderPlot({  
    hist(simData(), col = input$colInput)  
  })  
}
```



# Shiny Dashboard

- Package developed by RStudio for producing Dashboards with Shiny
- Available on github + CRAN

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



# What Next?

- This evening's LondonR meeting!
- EARL 2018 – 15% off for LondonR members, use code '**LONDONR**'



# LondonR tonight

- ***Now what? Integrating the output of your analysis with your organisation's infrastructure***

Jan Freyberg, ASI Data Science

- ***Inferring the effect of marketing campaigns using CausalImpact package***

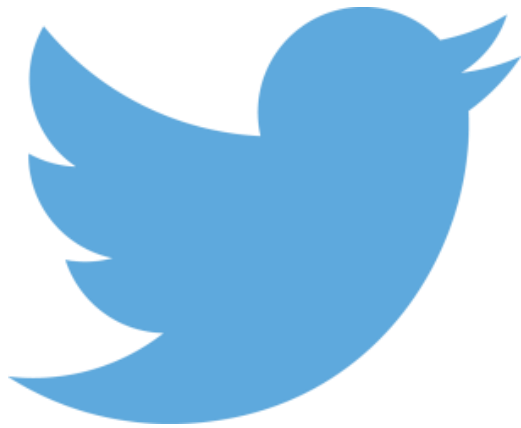
Ana Daglis, Farfetch

- ***good practice for R Packages***

Hannah Frick, Mango Solutions



# Follow Mango!



#ShinyAppreciation

@mangothecat

@earlconf

