Introduction to Shiny

LondonR – Workshop June 27th 2018

Nicolas Attalides
Data Scientist

nattalides@mango-solutions.com
WiFi

Network Name: UCLGuest
or use: 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:

Welcome to LondonR!

You entered the text: Welcome to LondonR!
Worked Example 1 – ui.R

```r
fluidPage(
  titlePanel("My First Shiny App!"),
  sidebarLayout(
    sidebarPanel(
      textInput("myText", "Enter text here:"),
    ),
    mainPanel(
      textOutput("niceTextOutput")
    )
  )
)
```
function(input, output) {
  output$niceTextOutput <- renderText({
    paste("You entered the text:", input$myText)
  })
}
Schematic of a Basic Shiny app

ui.R

```r
shinyUI(..
  textInput("TXT"),
  textOutput("text")
  ..)
```

server.R

```r
shinyServer(..
  output$text <- renderText("TXT")
  ..)
```

App

```
My First Shiny App!
```

```
Enter text here: "TXT"
```

```
You entered the text: Welcome to LondonR!
```

R
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
fluidPage(
  titlePanel("Title Here!")
)
}
sidebarLayout(
  sidebarPanel(
    #INPUTS GO HERE
  ),
  mainPanel(
    #OUTPUTS GO HERE
  )
)
)
function(input, output){
    #CODE GO HERE
}

server.R - Boiler Plate
## Input Controls

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>textInput()</code></td>
<td>Text string input</td>
</tr>
<tr>
<td><code>numericInput()</code></td>
<td>Numeric value input</td>
</tr>
<tr>
<td><code>selectInput()</code></td>
<td>Select single or multiple values from drop down list</td>
</tr>
<tr>
<td><code>sliderInput()</code></td>
<td>Numeric (single or range) “slider” input</td>
</tr>
<tr>
<td><code>radioButtons()</code></td>
<td>Set of radio button inputs</td>
</tr>
<tr>
<td><code>fileInput()</code></td>
<td>File upload control</td>
</tr>
</tbody>
</table>
Worked Example 2

My First Shiny App!

Enter text here:
Welcome to LondonR!

Select a number:
50

Select from the dropdown:
A
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

<table>
<thead>
<tr>
<th>Function</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>p()</td>
<td>A paragraph of text</td>
</tr>
<tr>
<td>h*()</td>
<td>A level <em>(1, 2, 3,...)</em> header</td>
</tr>
<tr>
<td>code()</td>
<td>A block of code</td>
</tr>
<tr>
<td>img()</td>
<td>An image</td>
</tr>
<tr>
<td>strong()</td>
<td>Bold text</td>
</tr>
<tr>
<td>em()</td>
<td>Italic text</td>
</tr>
</tbody>
</table>
My First Shiny App!

Enter text here:
Welcome to LondonR!

Select a number:
50

Select from the dropdown:
A

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
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

```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")
    )
  )
)
```
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")`
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")
    )
  )
)
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

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

<table>
<thead>
<tr>
<th>Ozone</th>
<th>Solar.R</th>
<th>Wind</th>
<th>Temp</th>
<th>Month</th>
<th>Day</th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>190</td>
<td>7.4</td>
<td>67</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>36</td>
<td>118</td>
<td>8</td>
<td>72</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>149</td>
<td>12.6</td>
<td>74</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>18</td>
<td>313</td>
<td>11.5</td>
<td>62</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>274</td>
<td>10.9</td>
<td>68</td>
<td>5</td>
<td>14</td>
</tr>
</tbody>
</table>
Worked Example 3 – ui.R

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

Histogram of mpg

Frequency

mpg

[Graph showing a histogram of mpg values]
Worked Example 4 – ui.R

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

```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

```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:
100

Select a colour:
yellow
red
yellow
blue

Histogram of simData()
Worked Example 5 – ui.R

```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:
100

Select a colour
red

Histogram of simData()
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")
    )
  )
)
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 CausallImpact package**
  Ana Daglis, Farfetch

- **good practice for R Packages**
  Hannah Frick, Mango Solutions
Follow Mango!

#ShinyAppreciation
@mangothecat
@earlconf