R Shiny Essentials: From Data to Interactive Dashboards

Kasia Koziara for QCIF workshop 2024

***

Before the workshop:

Prerequisites: recent versions of R and RStudio installed. Basic understanding of R programming and basic RStudio usage.

R packages to be installed:

- shiny
- shinythemes
- tidyverse

***

Data:

Link to Google Spreadsheet formatted workshop data:
https://tinyurl.com/4rpztktt

(can also share a .csv file via zoom, google doc, etc…)

Original data: https://www.data.brisbane.qld.gov.au/data/dataset/employment-projection-model
Employment Projection Model: A set of employment forecasts which reflect the Brisbane City Council view of the likely SEQ Regional Employment patterns in the period between 2011 and 2041. These were prepared by the National Institute of Economic and Industrial Research.
The .csv file used in this workshop was prepared based on the above datasets, it includes information about top 3 occupations and industries per number of residents in a suburb forecast for 2041.

***

***

Workshop instructions and code:

Note: New code is highlighted in each section.

1. **Get started:**
    a. Create a new project in R
    b. Copy and paste **bcc_occupation_industry.csv** into the working directory (project folder)
    c. Open a new R script
    d. Add the following at the top of the script:

```
# Read necessary packages
library(tidyverse)
library(shiny)
library(shinythemes)

# Create the user interface:
ui <- fluidPage()

# Create the server function:
server <- function(input, output) {}

# Combine them into an app:
shinyApp(ui = ui, server = server)
```

    e. Save the script as app.R
    f. Press 'Run app' option and see that blank page shows up in the server.

2. **Start building the UI**
    a. Add a title panel, and a sidebar/main panel layout

```
library(tidyverse)
library(shiny)
library(shinythemes)



# Create the user interface:
ui <- fluidPage(
  titlePanel("I am adding a title!"),
  sidebarLayout(
```

```
            sidebarPanel("put my widgets here"),
            mainPanel("put my outputs here")
            )
)

# Create the server:
server <- function(input, output) {}

# Combine them into an app:
shinyApp(ui = ui, server = server)
```

b. Save, run app again (you may need to stop “listening” - press stop in Console)

3. **Import *bcc_occupation_industry.csv***

a. Copy & paste the bcc_occupation_industry.csv file into your project folder (see that it shows up in your working directory in RStudio)

b. Under the library(packages) add a line of code (but before the UI) to read in the data as object **bcc**.

```
library(tidyverse)
library(shiny)
library(shinythemes)

bcc <- read_csv("bcc_occupation_industry.csv")

# Create the user interface:
ui <- fluidPage(
 titlePanel("I am adding a title!"),
 sidebarLayout(
            sidebarPanel("put my widgets here"),
            mainPanel("put my outputs here")
            )
)

# Create the server:
server <- function(input, output) {}

# Combine them into an app:
shinyApp(ui = ui, server = server)
```

4. Add your first widget in the side panel

```
library(tidyverse)
library(shiny)
library(shinythemes)
```

```
bcc <- read_csv("bcc_occupation_industry.csv")

ui <- fluidPage(
 titlePanel("I am adding a title!"),
 sidebarLayout(
        sidebarPanel("put my widgets here",
        selectInput(inputId = "suburb_select",
                    label = "Choose a suburb",
                    choices = unique(bcc$suburb)
        )
        ),
        mainPanel("put my outputs here")
 )
)

server <- function(input, output) {}
shinyApp(ui = ui, server = server)
```

a. Save & run your app, and see that a dropdown menu with the different suburbs appears

5. Build a reactive graph in the server, based on selections made in the 'species' widget

```
library(tidyverse)
library(shiny)
library(shinythemes)

bcc <- read_csv("bcc_occupation_industry.csv")

ui <- fluidPage(
 titlePanel("I am adding a title!"),
 sidebarLayout(
        sidebarPanel("put my widgets here",
        selectInput(inputId = "suburb_select",
                    label = "Choose a suburb",
                    choices = unique(bcc$suburb)
        )
        ),
        mainPanel("put my outputs here")
 )
)

server <- function(input, output) {

        suburb_industry <- reactive({
```

```
                bcc %>%
                filter(suburb == input$suburb_select) %>%
                select(industry, industry_count_2041)
        })

        output$industry_table <- renderTable({
                suburb_industry()
        })
}
shinyApp(ui = ui, server = server)
```

a) Try running the app again...notice that our table doesn't show up! That's because we haven't added it to our UI yet. We need to call our table back in the UI main panel:

```
library(tidyverse)
library(shiny)
library(shinythemes)

bcc <- read_csv("bcc_occupation_industry.csv")

ui <- fluidPage(
 titlePanel("I am adding a title!"),
 sidebarLayout(
        sidebarPanel("put my widgets here",
        selectInput(inputId = "suburb_select",
                        label = "Choose a suburb",
                        choices = unique(bcc$suburb)
        )
        ),
        mainPanel("put my outputs here",
                p("Suburb's top industries:"),
        tableOutput(outputId = "industry_table")
        )
 )
)

server <- function(input, output) {

        suburb_industry <- reactive({
                bcc %>%
                filter(suburb == input$suburb_select) %>%
                select(industry, industry_count_2041)
        })

        output$industry_table <- renderTable({
```

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

b) Save, run app again, industry_table shows up in the main panel.

6. Add another widget to add a graph of top occupations by *region*.

```
library(tidyverse)
library(shiny)
library(shinythemes)

bcc <- read_csv("bcc_occupation_industry.csv")

ui <- fluidPage(
 titlePanel("I am adding a title!"),
 sidebarLayout(
        sidebarPanel("put my widgets here",
        selectInput(inputId = "suburb_select",
                                label = "Choose a suburb:",
                                choices = unique(bcc$suburb)
                    ),
                            radioButtons(inputId = "region_select",
                                label = "Choose region:",
                                choices = unique(bcc$region))
        ),
        mainPanel("put my outputs here",
                p("Suburb's top industries:"),
        tableOutput(outputId = "industry_table")
        )
 )
)

server <- function(input, output) {

        suburb_industry <- reactive({
                bcc %>%
                filter(suburb == input$suburb_select) %>%
                select(industry, industry_count_2041)
        })

        output$industry_table <- renderTable({
                suburb_industry()
        })
```

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

a) Run the app - and notice that a second widget shows up, but the graph doesn't appear. We haven't created a reactive graph in the server, or called it back in the ui

7. In the server, make the reactive subset & graph based on the region_select input

```
library(tidyverse)
library(shiny)
library(shinythemes)

bcc <- read_csv("bcc_occupation_industry.csv")

ui <- fluidPage(
 titlePanel("I am adding a title!"),
 sidebarLayout(
        sidebarPanel("put my widgets here",
        selectInput(inputId = "suburb_select",
                                        label = "Choose a suburb:",
                                        choices = unique(bcc$suburb)
                ),
                                radioButtons(inputId = "region_select",
                                        label = "Choose region:",
                                        choices = unique(bcc$region))
        ),
        mainPanel("put my outputs here",
                p("Suburb's top industries:"),
        tableOutput(outputId = "industry_table")
        )
 )
)

server <- function(input, output) {

        suburb_industry <- reactive({
                bcc %>%
                filter(suburb == input$suburb_select) %>%
                select(industry, industry_count_2041)
        })

        output$industry_table <- renderTable({
                suburb_industry()
        })
```

```
    region_occupation <- reactive({
      bcc %>%
            filter(region == input$region_select) %>%
            count(occupation, occupation_rank_2041)
    })


    output$occupation_graph <- renderPlot({
      ggplot(region_occupation(), aes(x = occupation, y = n)) +
            geom_col(aes(fill = occupation_rank_2041)) +
            coord_flip() +
          scale_fill_manual(values = c("blue","gold","darkgreen")) +
    theme_minimal(base_size = 17)
    })
}
shinyApp(ui = ui, server = server)
```

a) Run the app - Why doesn't this graph show up? We haven't called it back to the ui:

```
library(tidyverse)
library(shiny)
library(shinythemes)

bcc <- read_csv("bcc_occupation_industry.csv")

ui <- fluidPage(
 titlePanel("I am adding a title!"),
 sidebarLayout(
        sidebarPanel("put my widgets here",
        selectInput(inputId = "suburb_select",
                                    label = "Choose a suburb:",
                                    choices = unique(bcc$suburb)
                    ),
                                radioButtons(inputId = "region_select",
                                    label = "Choose region:",
                                    choices = unique(bcc$region))
        ),
        mainPanel("put my outputs here",
                    p("Suburb's top industries:"),
        tableOutput(outputId = "industry_table"),
                    p("Region's top occupations:"),
                    plotOutput(outputId = "occupation_graph")
        )
 )
)
```

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

        suburb_industry <- reactive({
                bcc %>%
                filter(suburb == input$suburb_select) %>%
                select(industry, industry_count_2041)
        })

        output$industry_table <- renderTable({
                suburb_industry()
        })

        region_occupation <- reactive({
         bcc %>%
                filter(region == input$region_select) %>%
                count(occupation, occupation_rank_2041)
        })


        output$occupation_graph <- renderPlot({
         ggplot(region_occupation(), aes(x = occupation, y = n)) +
                geom_col(aes(fill = occupation_rank_2041)) +
                coord_flip() +
                scale_fill_manual(values = c("blue","gold","darkgreen")) +
                theme_minimal()
        })
}
shinyApp(ui = ui, server = server)
```

b) Run the app

8. Update the theme with shinythemes::shinytheme()

```
ui <- fluidPage(
 theme = shinytheme("cosmo"),
  titlePanel("I am adding
```

***

***

Some resources:

- Mastering Shiny by Hadley Wickham

- Building a Shiny app by Dean Attali
- Cheat sheet: https://raw.githubusercontent.com/rstudio/cheatsheets/main/shiny.pdf

- Useful tutorials and examples from RStudio: https://shiny.rstudio.com/tutorial/
- Cool examples: https://shiny.rstudio.com/gallery/
- widget options

- Sharing your Shiny app: https://shiny.rstudio.com/tutorial/written-tutorial/lesson7/
    - If at UQ, there is a server to host Shiny apps: https://shiny.rcc.uq.edu.au/
        - Please contact: Dr David Green, HPC Manager, Research Computing Centre (david.green@uq.edu.au)

***