Data Visualization

Getting Fancy: Interactive Data Visualization

Ciara Zogheib

Data Sciences Institute, University of Toronto

Exit Survey Time!

Assignment 4 Tip – Importing Data

- Toronto Open Data only (sorry!)
- Load datasets directly into R (without downloading as CSV/XLSX) using the related R package

```
library(opendatatoronto)
list_package_resources("https://open.toronto.ca/dataset/neighbourhoods/") |>
  head(1) |>
  get_resource()
```

In today's class, we will...

- Explore examples of dynamic data visualization
- Discuss the costs, benefits, and design considerations for creating dynamic data visualizations
- Develop basic understanding of rAmCharts, gganimate, and Shiny packages for creating dynamic data visualizations in R
- Review course content
- Work on (/ask questions about) Assignment 4

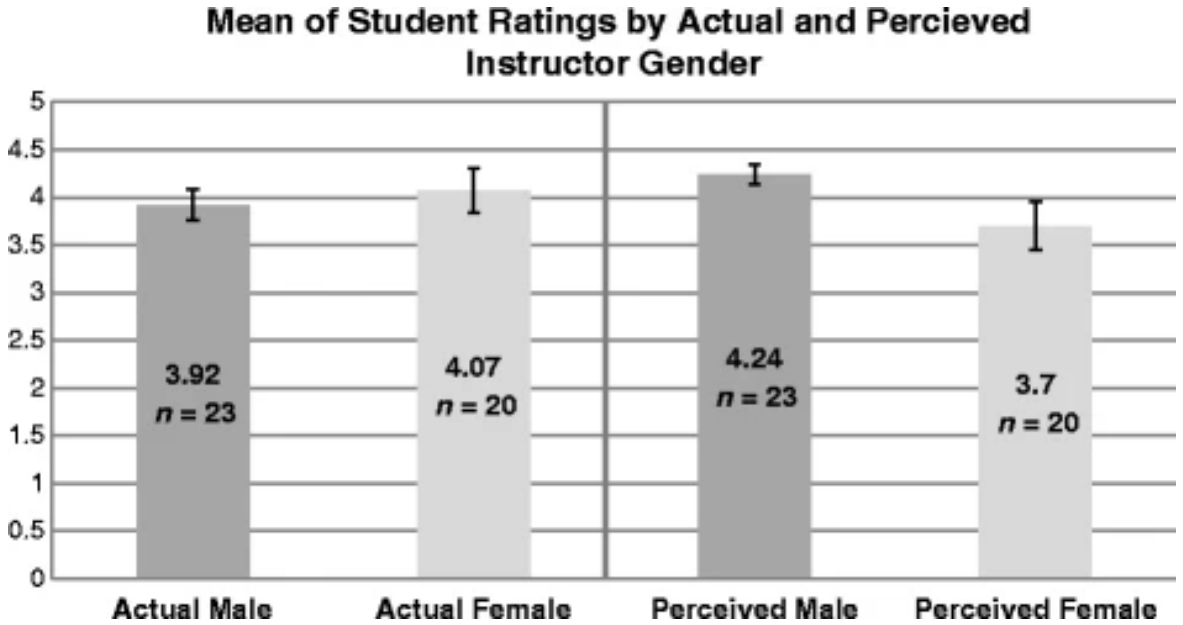
Case Study: Gender bias in teaching evaluations

How does faculty gender impact course evaluations?

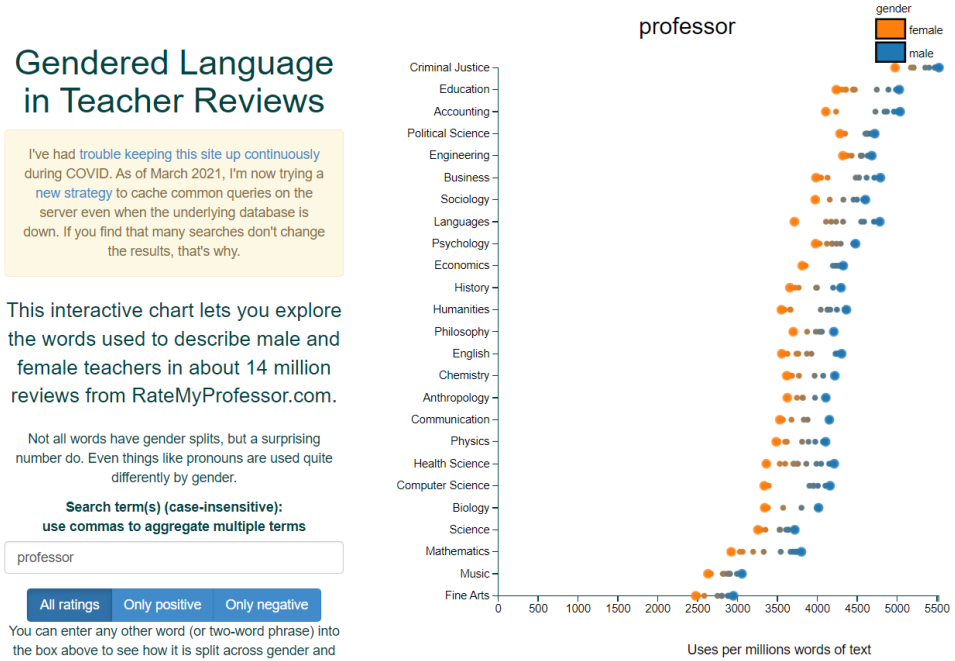
- The gender of instructors (perceived or actual) influences how they are scored by students on teaching and course evaluations, with student evaluations tending to be significantly biased in favour of men and against women, even when all else is equal
- The bias in these student evaluations can adversely impact the ability of female scholars to be “full-time tenure-track, to hold tenured positions, to attain higher leadership roles in academia, and to earn the same salary as males in the same positions”

(Chavez & Mitchell, 2020; MacNell et al., 2015)

Mean of Student Ratings by Actual and Perceived Instructor Gender (MacNell et al., 2015)

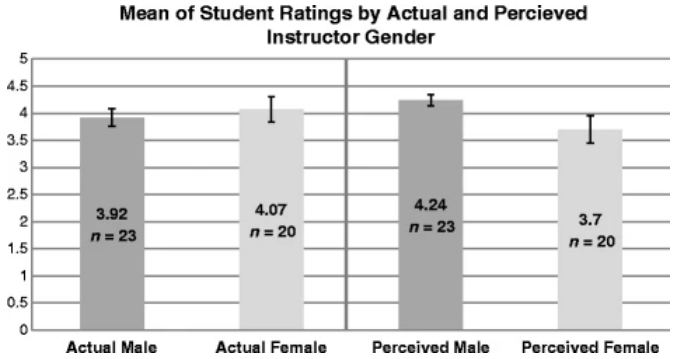


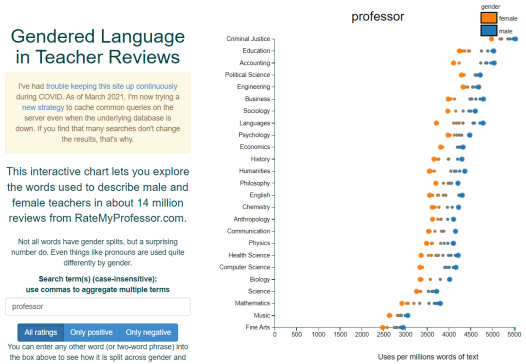
Gendered Language in Teacher Reviews (Schmidt, 2015)



(Click to go to dynamic visualization)

Activity: Comparing data visualizations





- **Let's discuss the two examples.**
  - How are they different? What does each visualization 'do'?
  - What are the pros and cons of each?

Static vs. dynamic data visualization

Defining static vs. dynamic data visualization

- Static data visualization
  - An image-based chart or infographic (think PDF, PNG, JPG)
  - A snapshot of data
  - Most of what we have seen so far in this course
- Dynamic data visualization
  - Interactive applications or web pages that allow users to modify or filter a data visualization
  - Multiple data stories in one

Benefits of dynamic data visualizations

- Dynamic data visualizations
  - Can provide information that cannot be obtained from static charts
  - Are useful for viewing individual-level data
  - Allow audiences to explore the data in-depth, supporting transparency and reproducibility
  - Can increase interest and engagement in research outputs

(Weissgerber et al., 2016)

Costs of dynamic data visualizations

- Dynamic data visualizations
  - Make it more challenging to tell a single clear story or communicate a clear message
  - Can be confusing or overwhelming for audiences
  - Present access and sharing challenges (e.g. an image can be viewed in print, online, on mobile, but a dynamic viz might need to be hosted on a particular platform or software)

Designing dynamic data visualizations

Considerations

- Best practices for data visualization and accessibility still apply, but we also need to consider unique elements of dynamic data visualizations. For example:
  - Are the interactive elements easy to navigate?
  - Are the interactive elements accessible?
  - How much time will your audience have to interact with the data visualization?
  - **Important: Do interactive features actually help your data visualization serve your purpose?**

Elements of dynamic data visualizations

- We can conceptualize changes to our dynamic data visualizations in terms of how they affect two visual elements of our plots:
  - Spatial elements → Position and quantity (e.g. number of data points on a plot, scales of axes)
  - Retinal elements → Size, brightness, rotation, patterning, shape, and colour
- **We need to consider:** in our dynamic data visualizations, will spatial and retinal elements be fixed or mutable? Can they be created and deleted based on user interaction? Can their meanings change?

(Cottam et al., 2012)

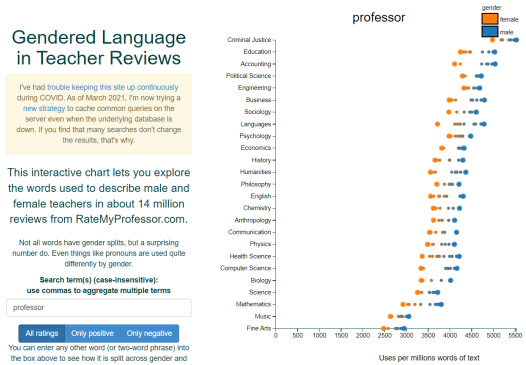
Types of changes

Identity-preserving	• Maintain associations between visual elements and underlying data • Some part of a representation stays constant (eg. keeping the relative position of data points constant) • Important for comparing across time/snapshots
Transitional	• Maintain some associations, and limit changes to known values • New elements may be added or changed • Balance flexibility with the ability to compare over short period of time/small changes between snapshots
Immediate	• Generally do not preserve associations • Create and delete elements, alter scales, remaps variables every time a new 'snapshot' is created • Does not allow for comparison across time/snapshots

(Cottam et al., 2012)

Activity: Types of changes

- Return to our earlier example of gendered language from RateMyProfessor.com
- What types of changes occur when we interact with this dynamic data visualization?
- Which spatial and retinal elements stay constant? Which change?
- If you were redesigning this viz, would you choose to make different changes? Why or why not?



Tools for dynamic data visualization

Tools for dynamic data visualizations

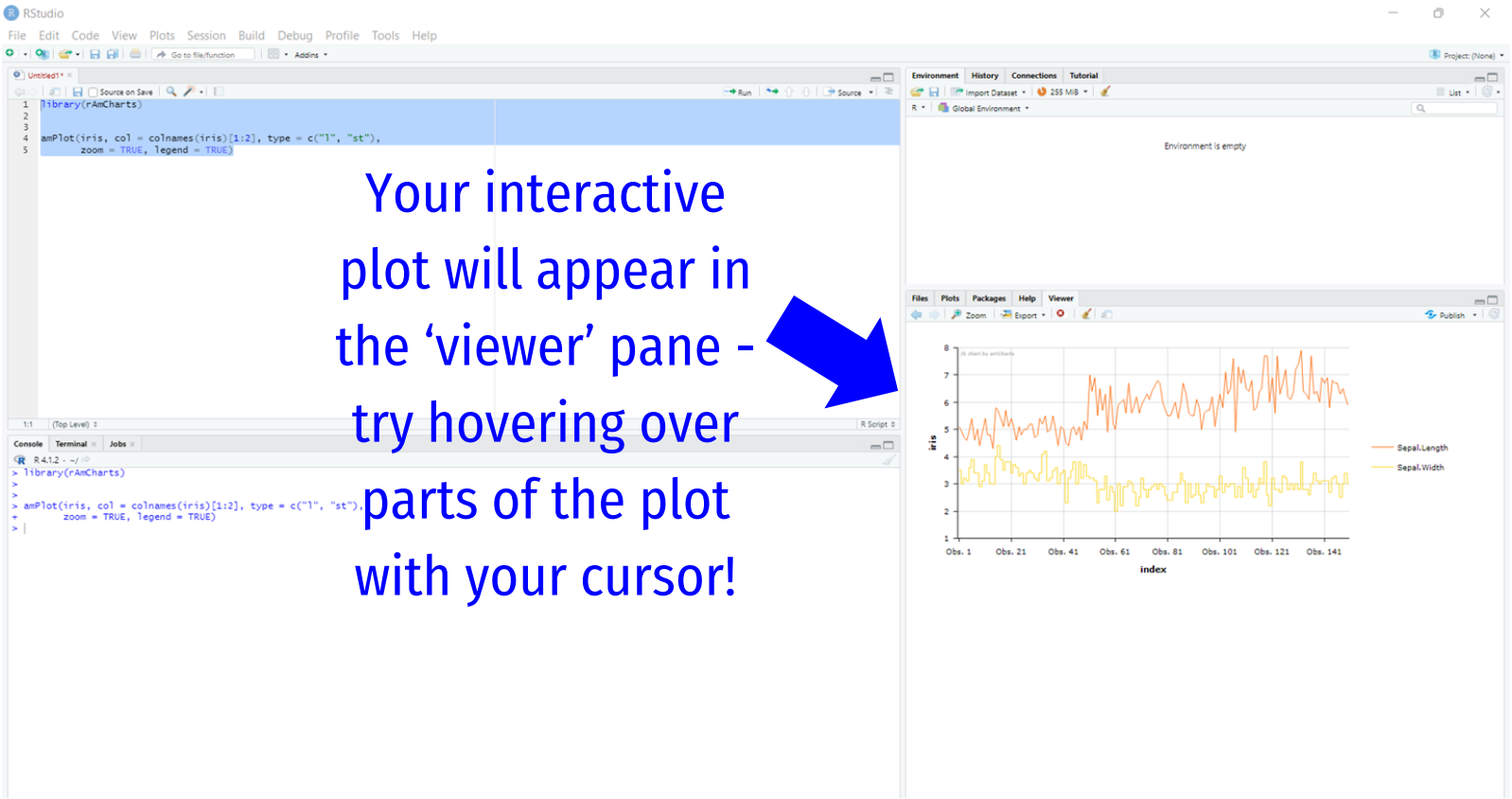
rAmCharts

- One of the simplest ways to create a dynamic data visualization in R is to use the `rAmCharts` library
- We can experiment using the iris sample dataset:

```
library(rAmCharts)

amPlot(iris, col = c('Sepal.Length', 'Sepal.Width’),

       type = c("line", "step"),

        zoom = TRUE, legend = TRUE)
```

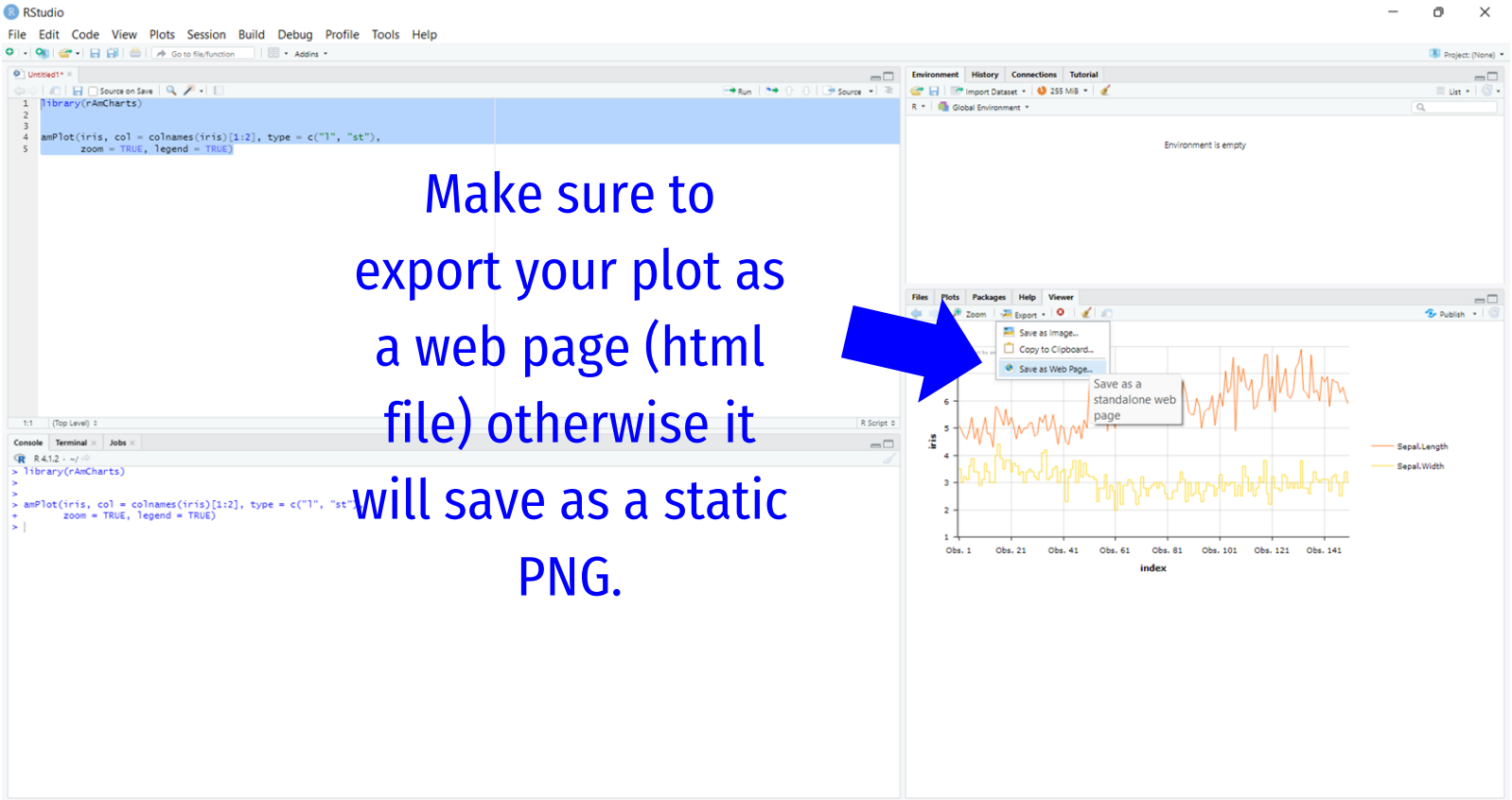
(Rouvière, 2021)

rAmCharts



Your interactive plot will appear in the 'viewer' pane - try hovering over parts of the plot with your cursor!

rAmCharts



Make sure to export your plot as a web page (html file) otherwise it will save as a static PNG.

gganimate

- `rAmCharts` uses base R graphing functionality, but if we want to use `ggplot`, we can play with the `gganimate` add-on and `gifski` renderer
- To demonstrate, we will use our `gapminder` sample dataset

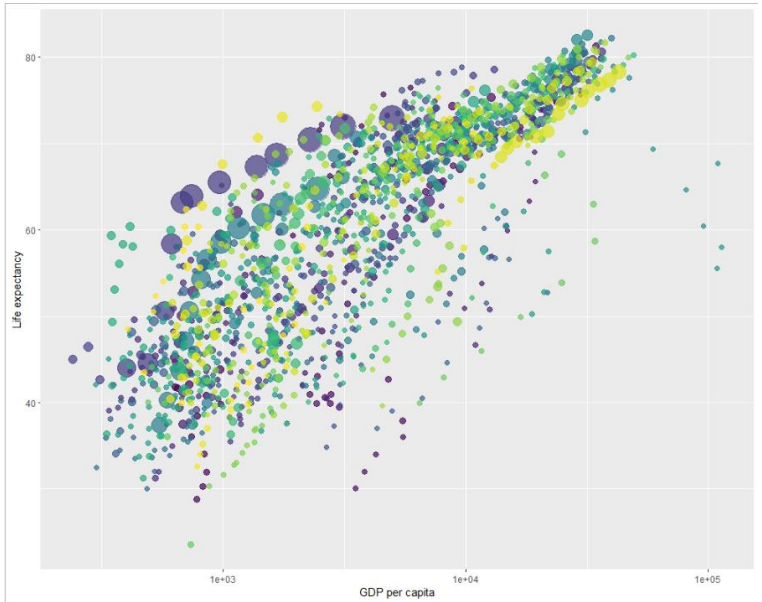
```
library(ggplot2)

library(gganimate)

library(gifski)

library(gapminder)
```

gganimate

- First, we make a static ggplot: a scatterplot showing the relationship between GDP and life expectancy

```
p <- ggplot(gapminder, aes(x = gdpPercap, y=lifeExp, size = pop, colour = country)) +
  geom_point(show.legend = FALSE, alpha = 0.7) +
  scale_color_viridis_d() +
  scale_size(range = c(2, 12)) +
  scale_x_log10() +
  labs(x = "GDP per capita", y = "Life expectancy")
p
```

gganimate




```
p <- ggplot(gapminder, aes(x = gdpPercap, y=lifeExp, size = pop, colour = country)) + geom_point(show.legend = FALSE, alpha = 0.7) + scale_color_viridis_d() + scale_size(range = c(2, 12)) + scale_x_log10() + labs(x = "GDP per capita", y = "Life expectancy")
```

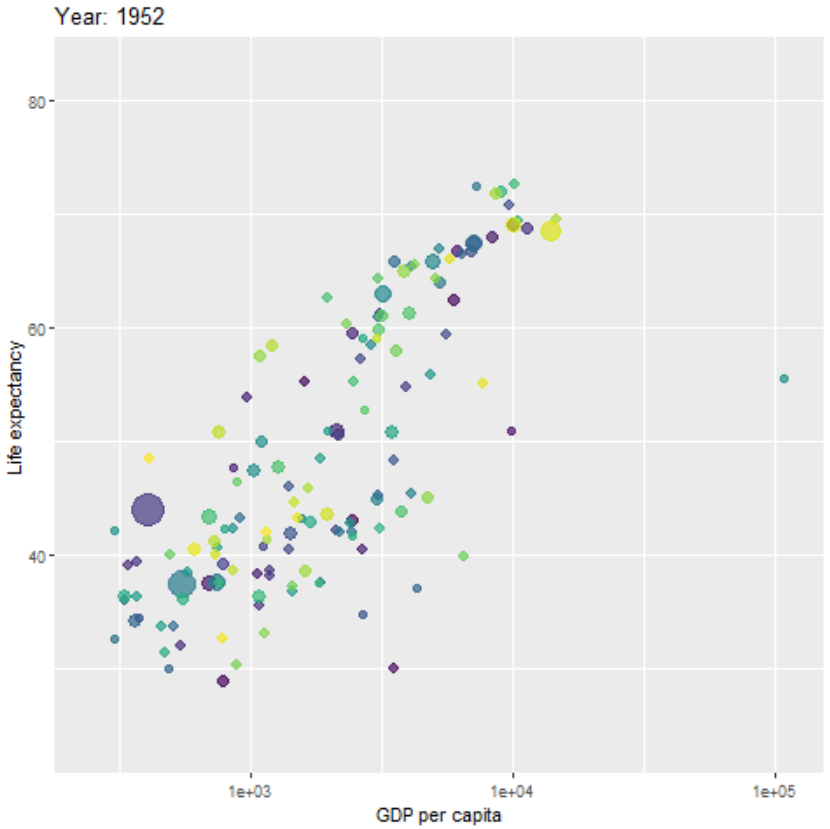
gganimate

- If we want to show how our plot, p, changes over time, we use the `transition_time()` function, with our time variable as the argument
- The `frame_time` variable in our label argument ensures that our label will change along with the plot

```
p2 <- p + transition_time(year) +
  labs(title = "Year: {frame_time}")
animate(p2, duration = 10, renderer = gifski_renderer())
```

gganimate

```
p2 <- p + transition_time(year) +

  labs(title = "Year:
{frame_time}")

animate(p2,duration = 10,renderer
= gifski_renderer())
```



Shiny

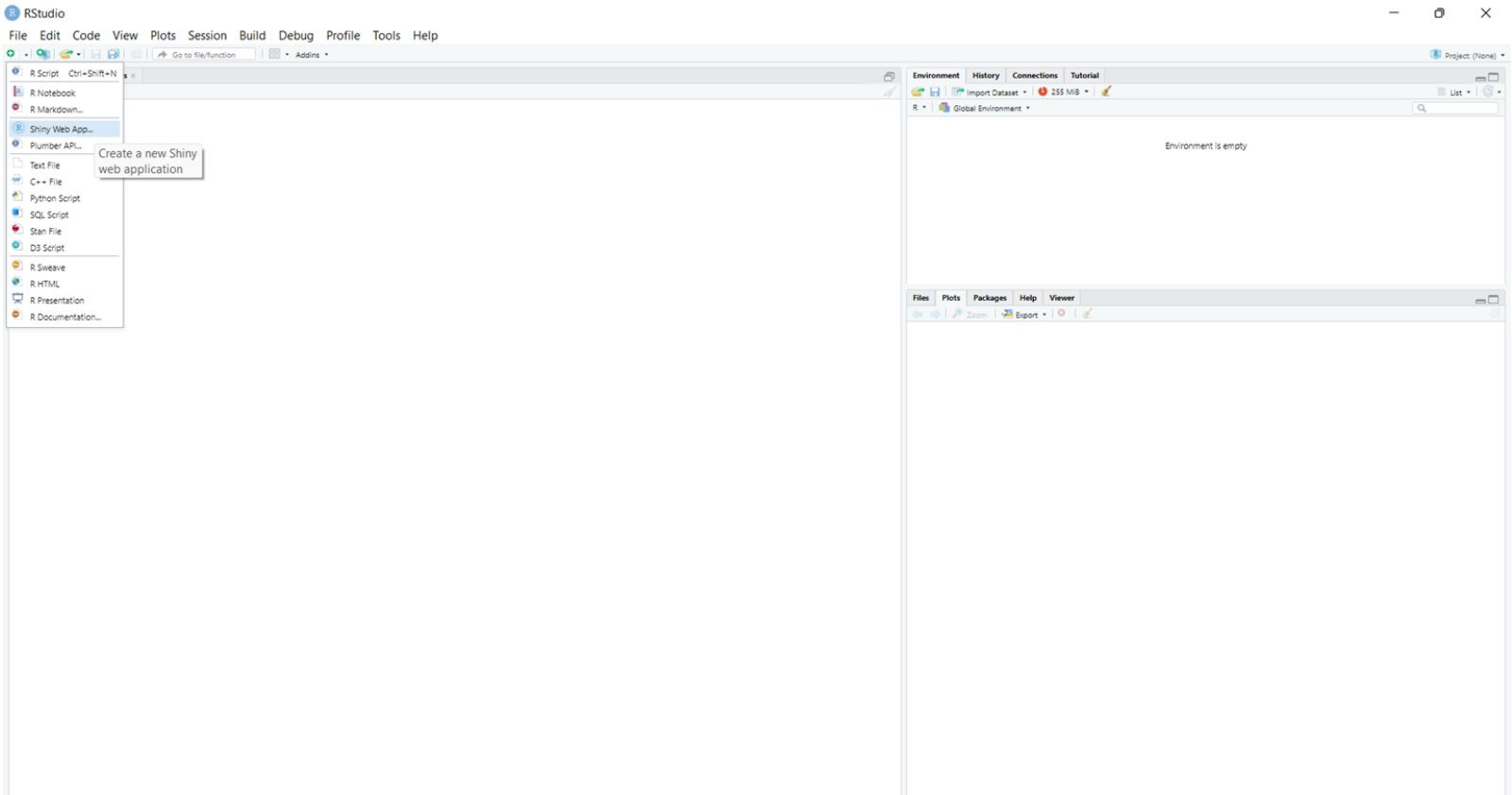
- Shiny is a package that allows us to use R language to code web applications
- We can use Shiny to create dynamic data visualizations
- A Shiny app has three components:
  - user interface object
  - server function
  - call to the shinyApp function

(RStudio, 2020)

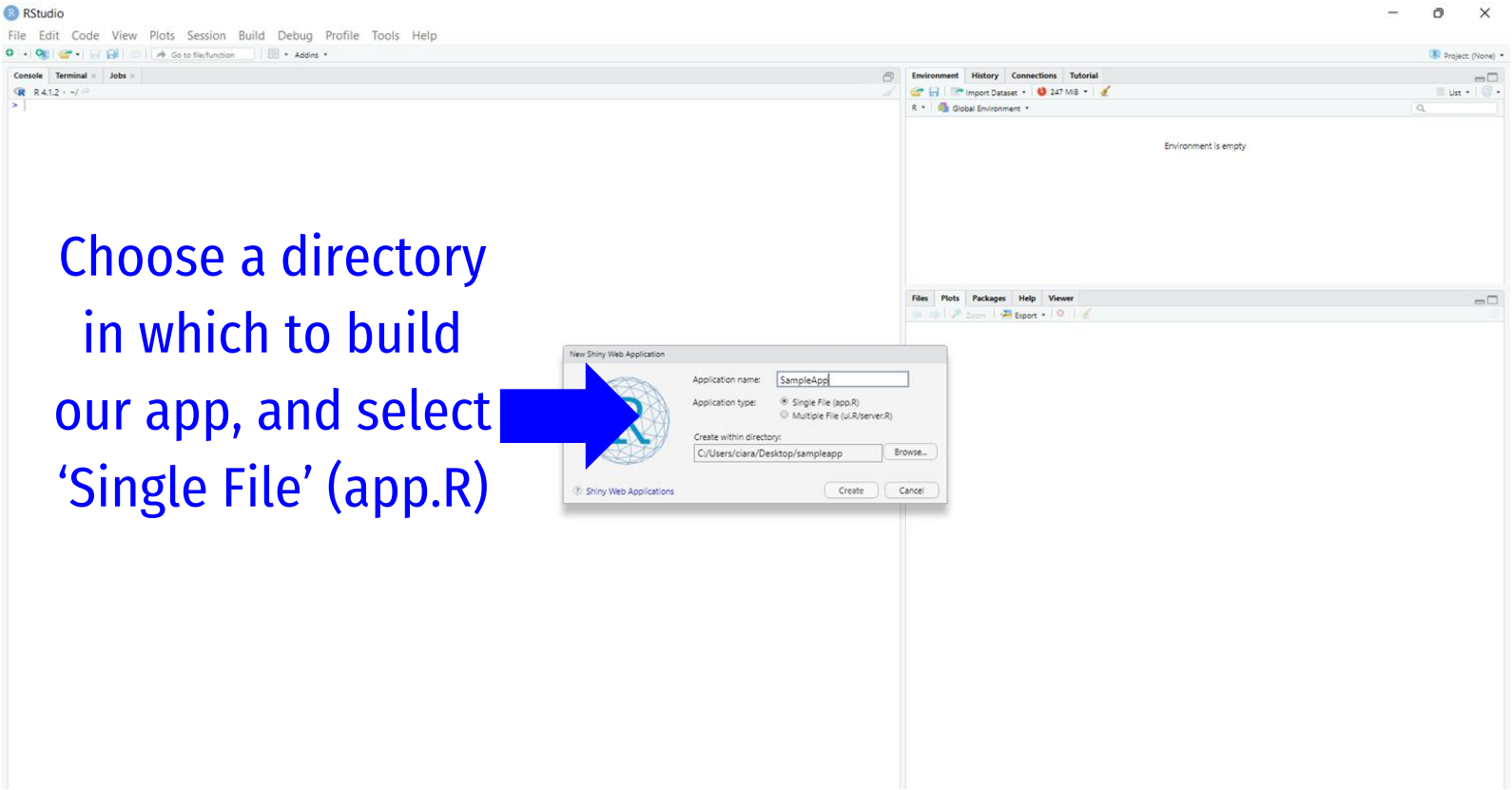
First: Install shiny

```
install.packages("shiny")
```

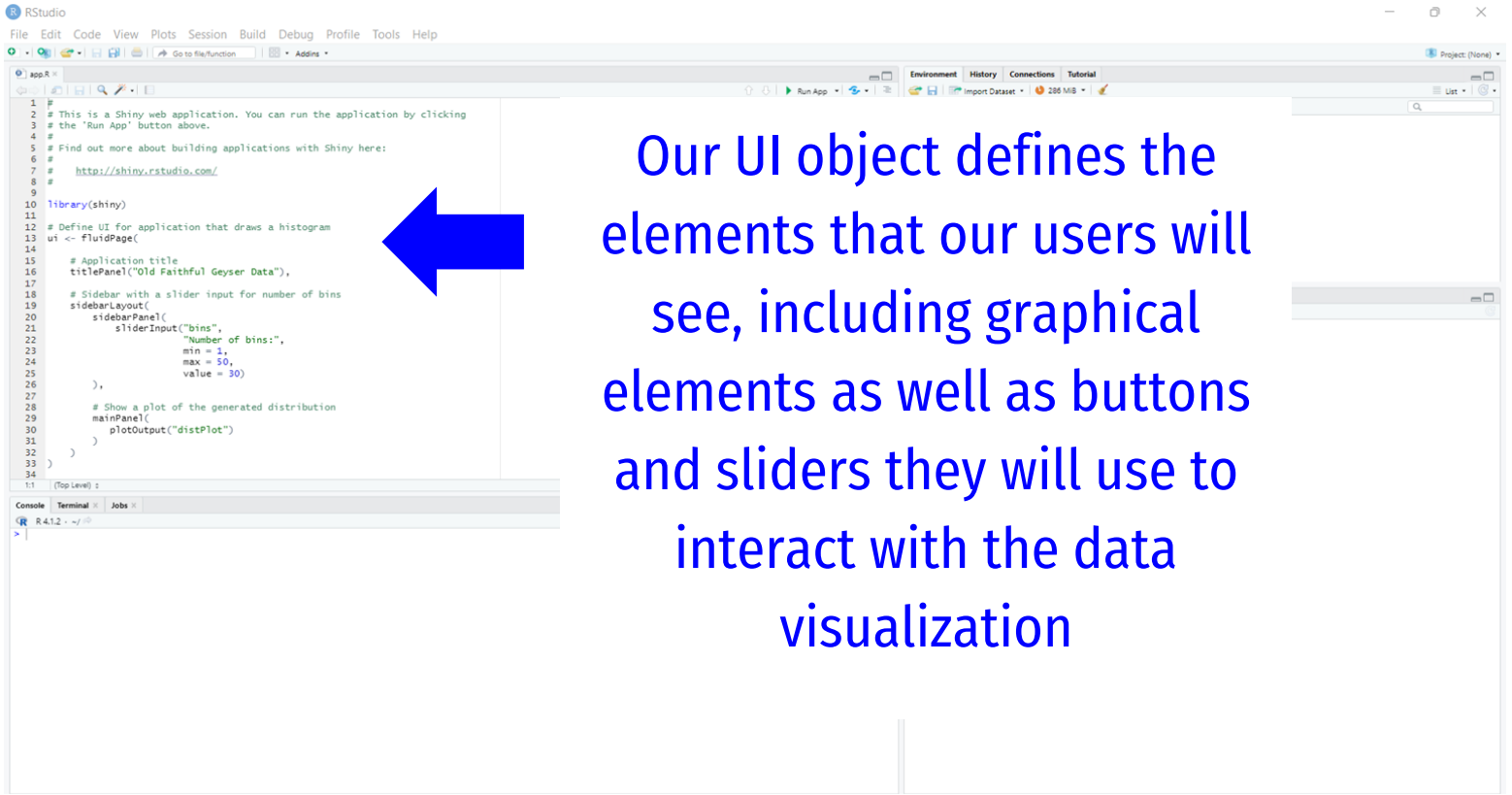
Shiny


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
R Script Ctrl+Shift+N
R Notebook
R Markdown...
Shiny Web App...
Plumber API...
Text File
C++ File
Python Script
SQL Script
Stan File
D3 Script
R Sweave
R HTML
R Presentation
R Documentation...
Create a new Shiny web application
Environment History Connections Tutorial
Import Dataset
255 MiB
List
R
Global Environment
Environment is empty
Files Plots Packages Help Viewer
Zoom
Export

Shiny

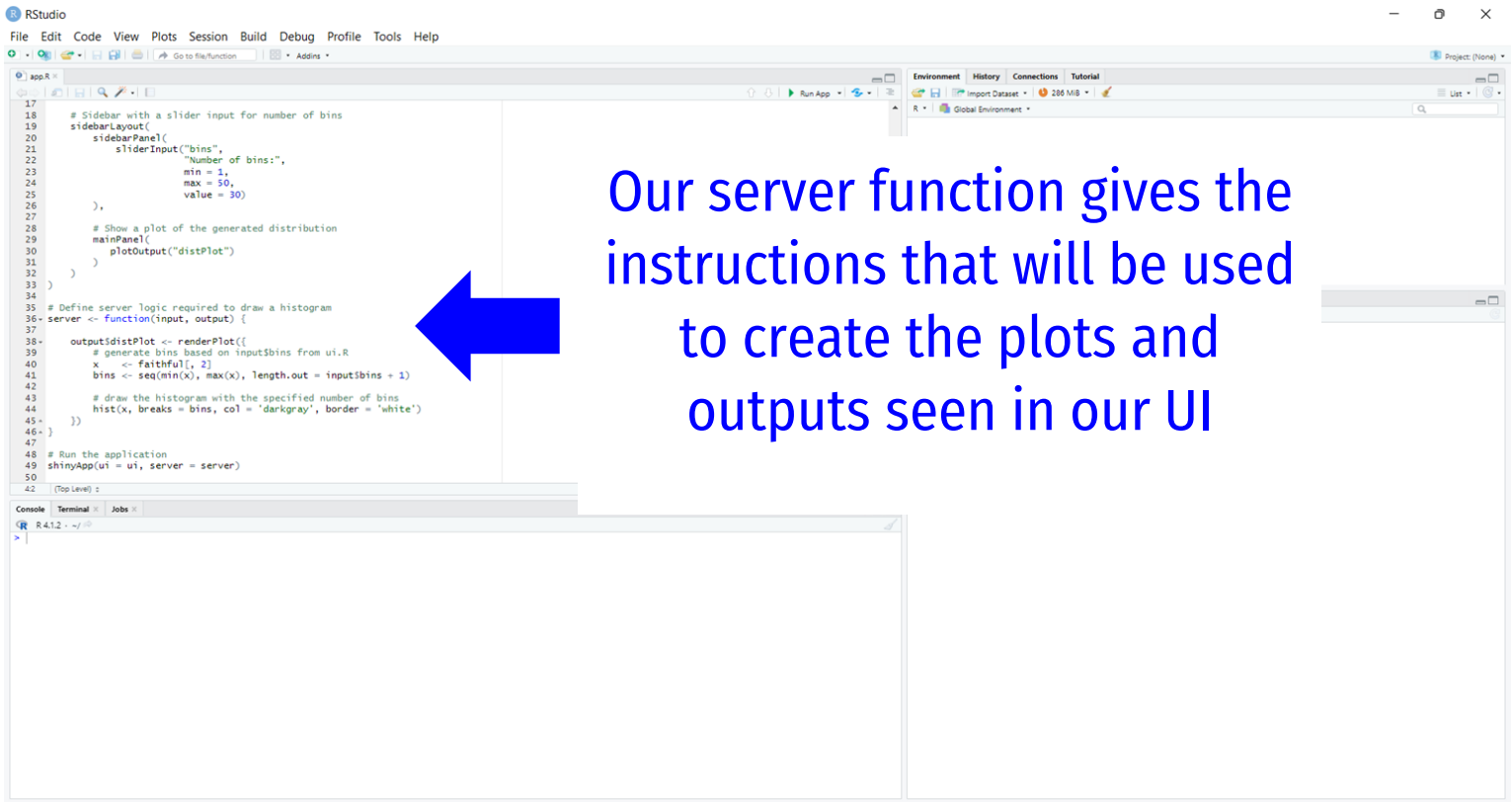

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Console Terminal Jobs
Environment History Connections Tutorial
Environment is empty
Files Plots Packages Help Viewer
Choose a directory
in which to build
our app, and select
'Single File' (app.R)
New Shiny Web Application
Application name: SampleApp
Application type: Single File (app.R)
Multiple File (ui.R/server.R)
Create within directory:
C:/Users/ciara/Desktop/sampleapp
Browse...
Shiny Web Applications
Create
Cancel

Shiny



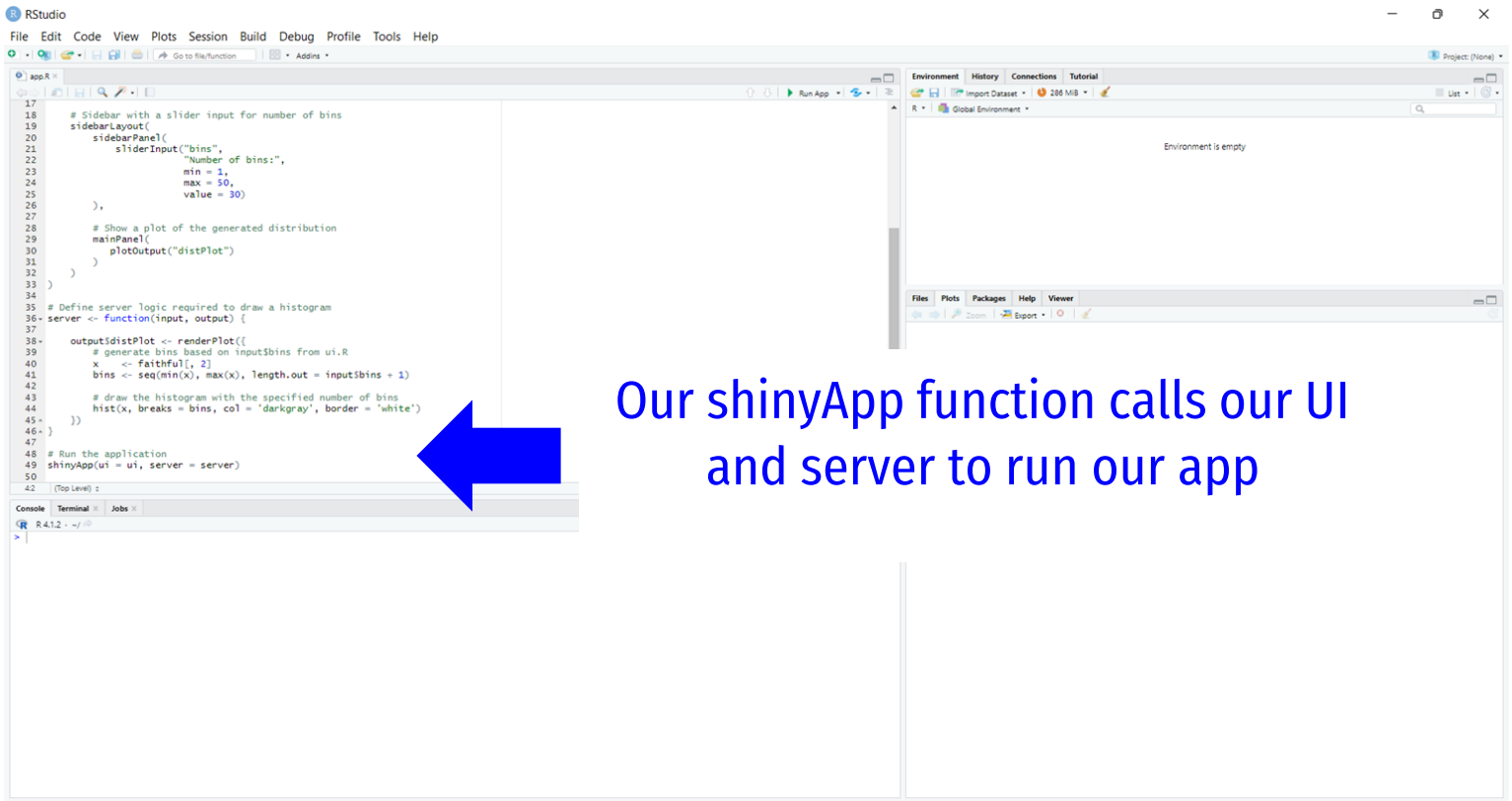
Our UI object defines the elements that our users will see, including graphical elements as well as buttons and sliders they will use to interact with the data visualization

Shiny



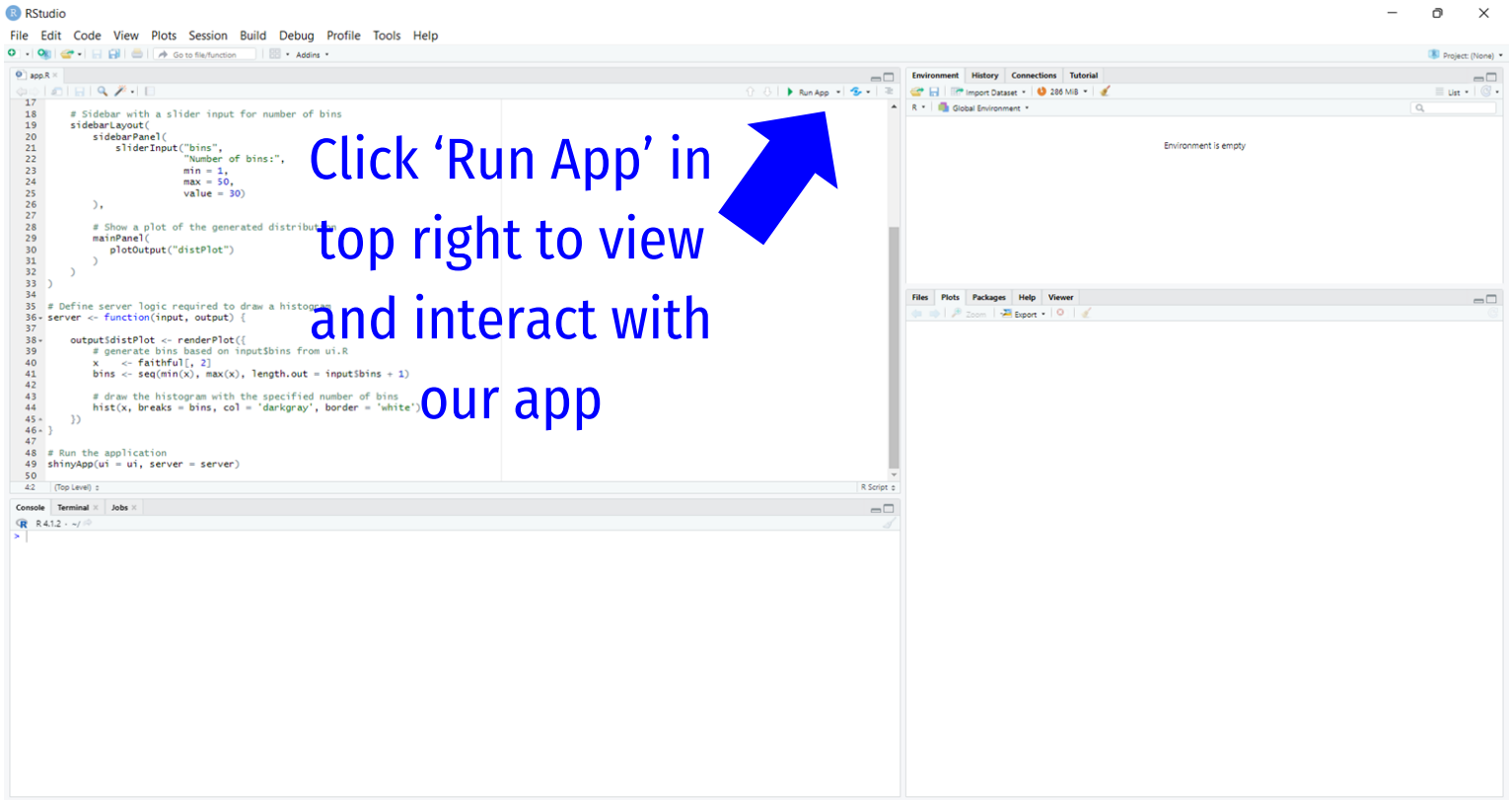

```r
    # Sidebar with a slider input for number of bins
    sidebarLayout(
        sidebarPanel(
            sliderInput("bins",
                        "Number of bins:",
                        min = 1,
                        max = 50,
                        value = 30)
        ),

        # Show a plot of the generated distribution
        mainPanel(
           plotOutput("distPlot")
        )
    )
)

# Define server logic required to draw a histogram
server <- function(input, output) {

    output$distPlot <- renderPlot({
        # generate bins based on input$bins from ui.R
        x    <- faithful[, 2]
        bins <- seq(min(x), max(x), length.out = input$bins + 1)

        # draw the histogram with the specified number of bins
        hist(x, breaks = bins, col = 'darkgray', border = 'white')
    })
}

# Run the application
shinyApp(ui = ui, server = server)
```

Shiny

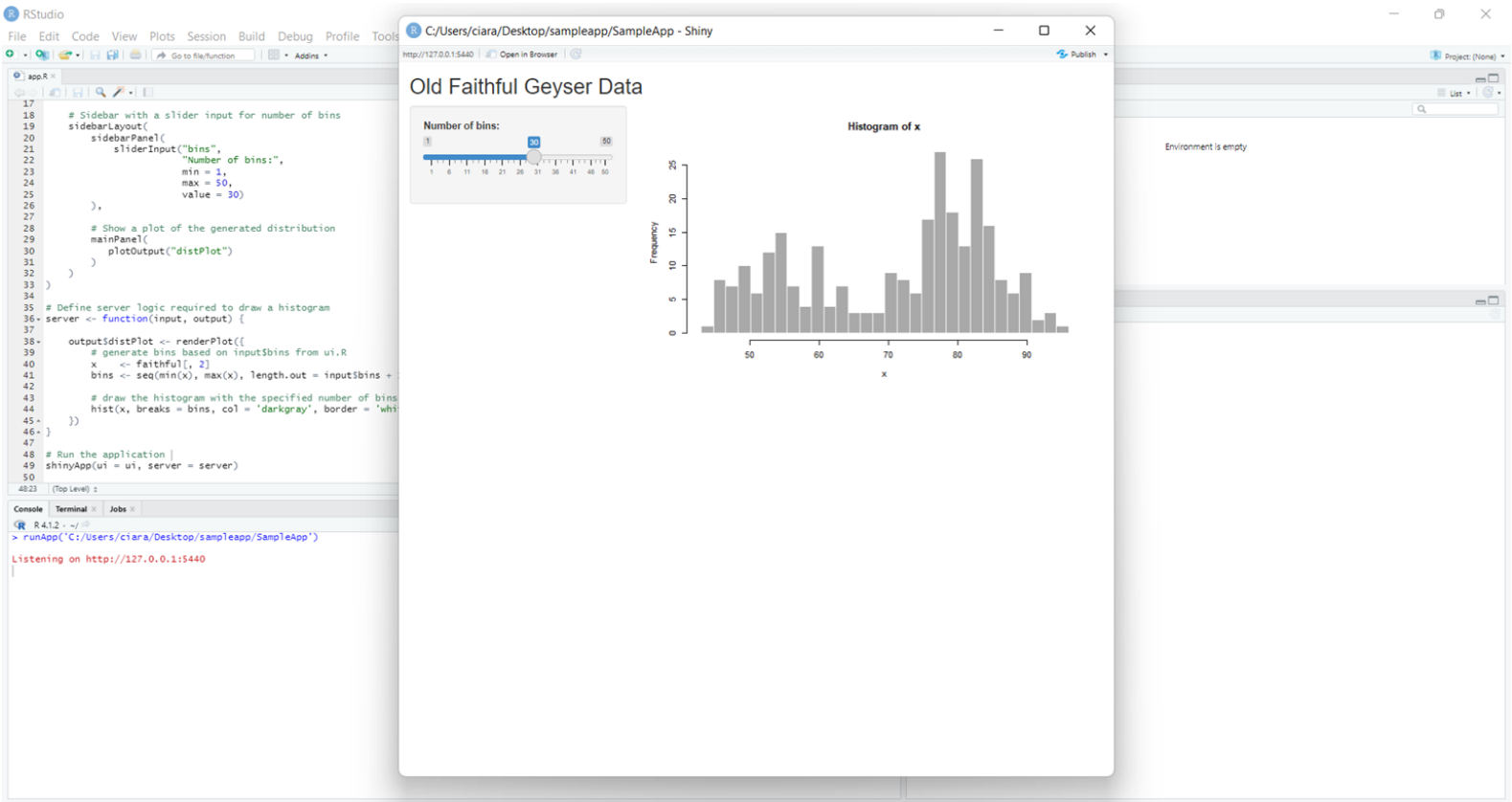



```
    # Sidebar with a slider input for number of bins
    sidebarLayout(
        sidebarPanel(
            sliderInput("bins",
                        "Number of bins:",
                        min = 1,
                        max = 50,
                        value = 30)
        ),

        # Show a plot of the generated distribution
        mainPanel(
           plotOutput("distPlot")
        )
    )
)

# Define server logic required to draw a histogram
server <- function(input, output) {

    output$distPlot <- renderPlot({
        # generate bins based on input$bins from ui.R
        x    <- faithful[, 2]
        bins <- seq(min(x), max(x), length.out = input$bins + 1)

        # draw the histogram with the specified number of bins
        hist(x, breaks = bins, col = 'darkgray', border = 'white')
    })
}

# Run the application
shinyApp(ui = ui, server = server)
```

Shiny


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
app.R
Run App
Click 'Run App' in top right to view and interact with our app
Environment History Connections Tutorial
Environment is empty
Files Plots Packages Help Viewer
Console Terminal Jobs

Shiny


RStudio
File Edit Code View Plots Session Build Debug Profile Tools
app.R
Sidebar with a slider input for number of bins
sidebarLayout(
sidebarPanel(
sliderInput("bins",
"Number of bins:",
min = 1,
max = 50,
value = 30)
),
Show a plot of the generated distribution
mainPanel(
plotOutput("distPlot")
)
)
)
# Define server logic required to draw a histogram
server <- function(input, output) {
output$distPlot <- renderPlot({
# generate bins based on input$bins from ui.R
x <- faithful[, 2]
bins <- seq(min(x), max(x), length.out = input$bins +
# draw the histogram with the specified number of bins
hist(x, breaks = bins, col = 'darkgray', border = 'whi
})
}
# Run the application
shinyApp(ui = ui, server = server)
Console Terminal Jobs
> runApp('C:/Users/ciara/Desktop/sampleapp/SampleApp')
Listening on http://127.0.0.1:5440
C:/Users/ciara/Desktop/sampleapp/SampleApp - Shiny
Open in Browser
Publish
Old Faithful Geyser Data
Number of bins:
Histogram of x
Frequency
x
Environment is empty

Sharing Shiny apps

- Shiny apps can be published at shinyapps.io, where they are accessible to anyone with the URL
- Once hosted at shinyapps.io, our interactive data visualizations can be embedded in websites, used as presentation aids, or shared with the public!

(RStudio, 2020)

Review Time!

Course Topics

- **First Steps**
  - Get started
  - Make a plot (substantive qualities)
  - Thinking about reproducibility
- **Graphing Our Data**
  - Show the right numbers (subsets, changing axes)
  - Graph tables, add labels
  - Choosing the right visualization (perceptual qualities)

Course Topics

- **Visualization with Purpose**
  - Refine our plots (aesthetic qualities)
  - Colour theory and accessible design
  - Data visualization as advocacy
- **Getting Fancy**
  - Working with models
  - Drawing maps and flow charts
  - Interactive data visualizations


school kahoot
games are more competitive
that the super bowl
CHANGE MY MIND

Assignment 4

Congratulations!

You have finished your intro to Data Visualization!

Learning Objectives of this Course

1. Develop ability to **create and customize data visualizations** start to finish in R
2. Build an understanding of general design principles for creating **accessible and equitable** data visualizations in R and other software
3. Build an understanding of **data visualization as purposeful/telling a story** (and the ethical/professional implications thereof)