# R:Shiny

網絡資料分析與模式

2021.04.30

杜承軒

## 期中考

平均值：68.8、標準差：17.1

公布於ceiba學習成績

評語：X X , 5 5 5 X 5 X , 5 5 X , 5 5 X (X:10分)

1.問答題　2.計算題　3-1.實作題　3-2.實作題

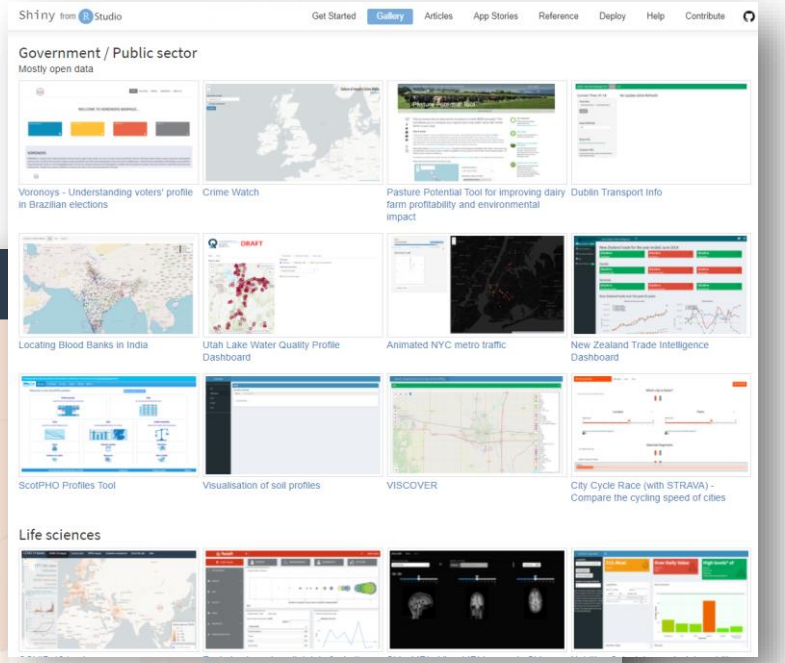
對成績有疑問，請在5/6(四)前寄信提出

## 期末報告計畫書

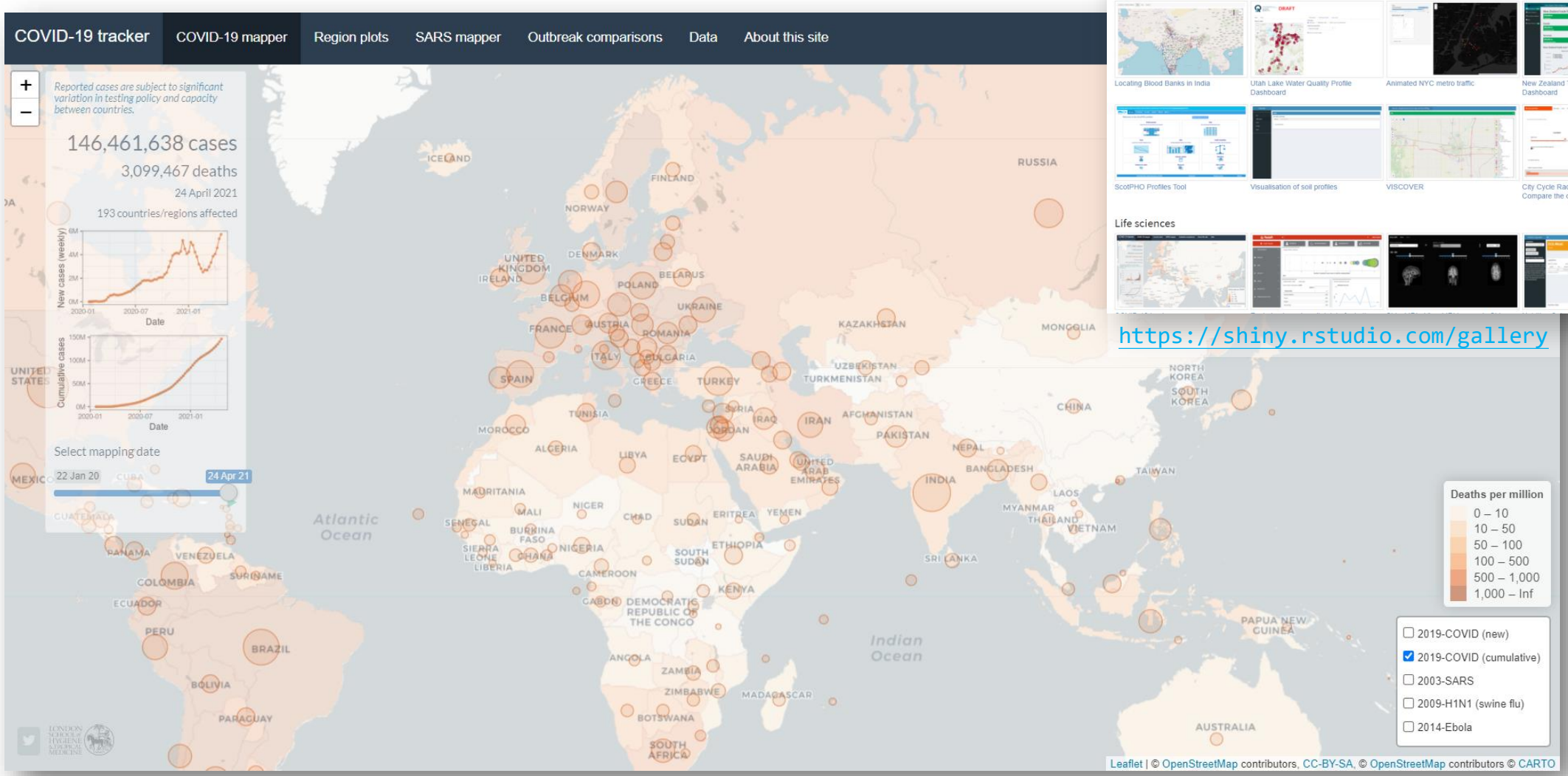
5/7 (五) 9:00 前繳交ceiba作業區（團體繳交）

格式與頁數不限

# Interactive Data Visualization in R



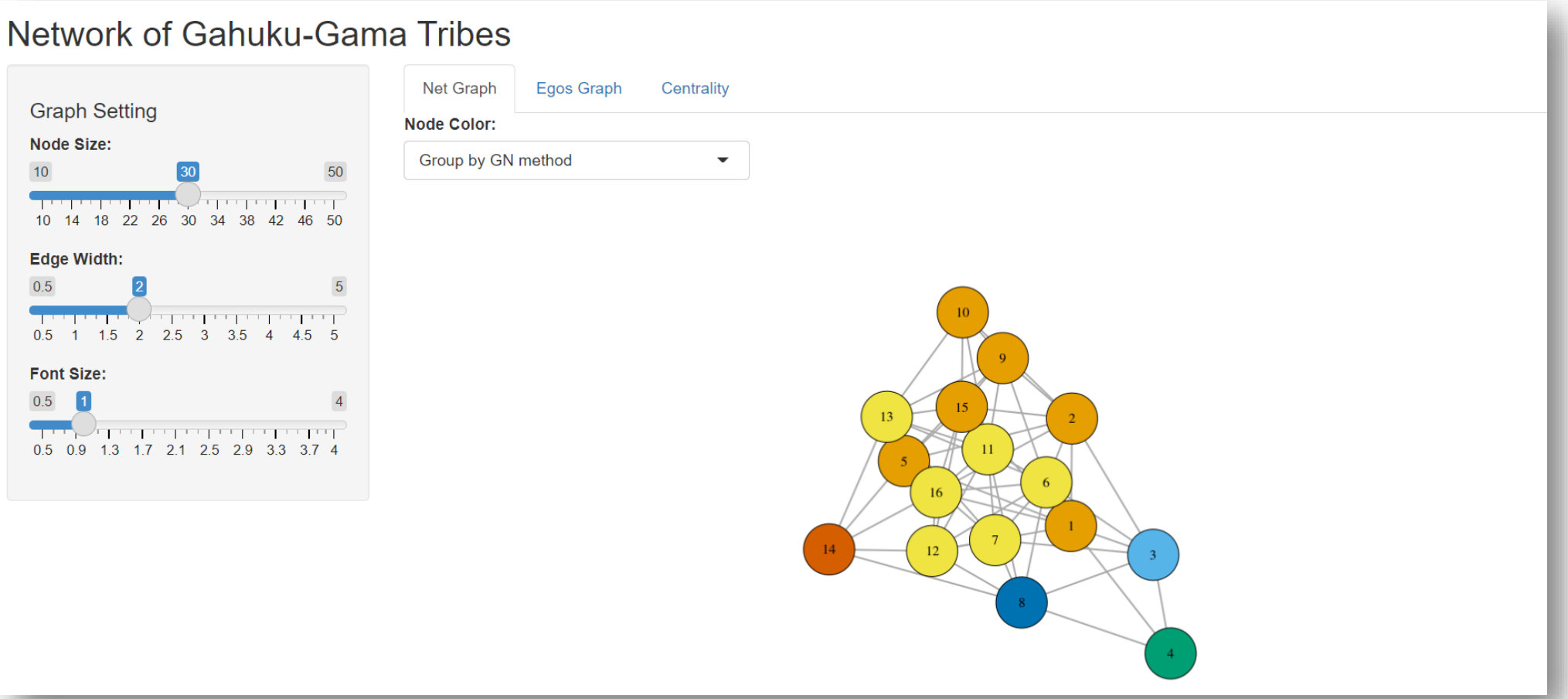

https://shiny.rstudio.com/gallery




COVID-19 tracker https://vac-lshtm.shinyapps.io/ncov_tracker

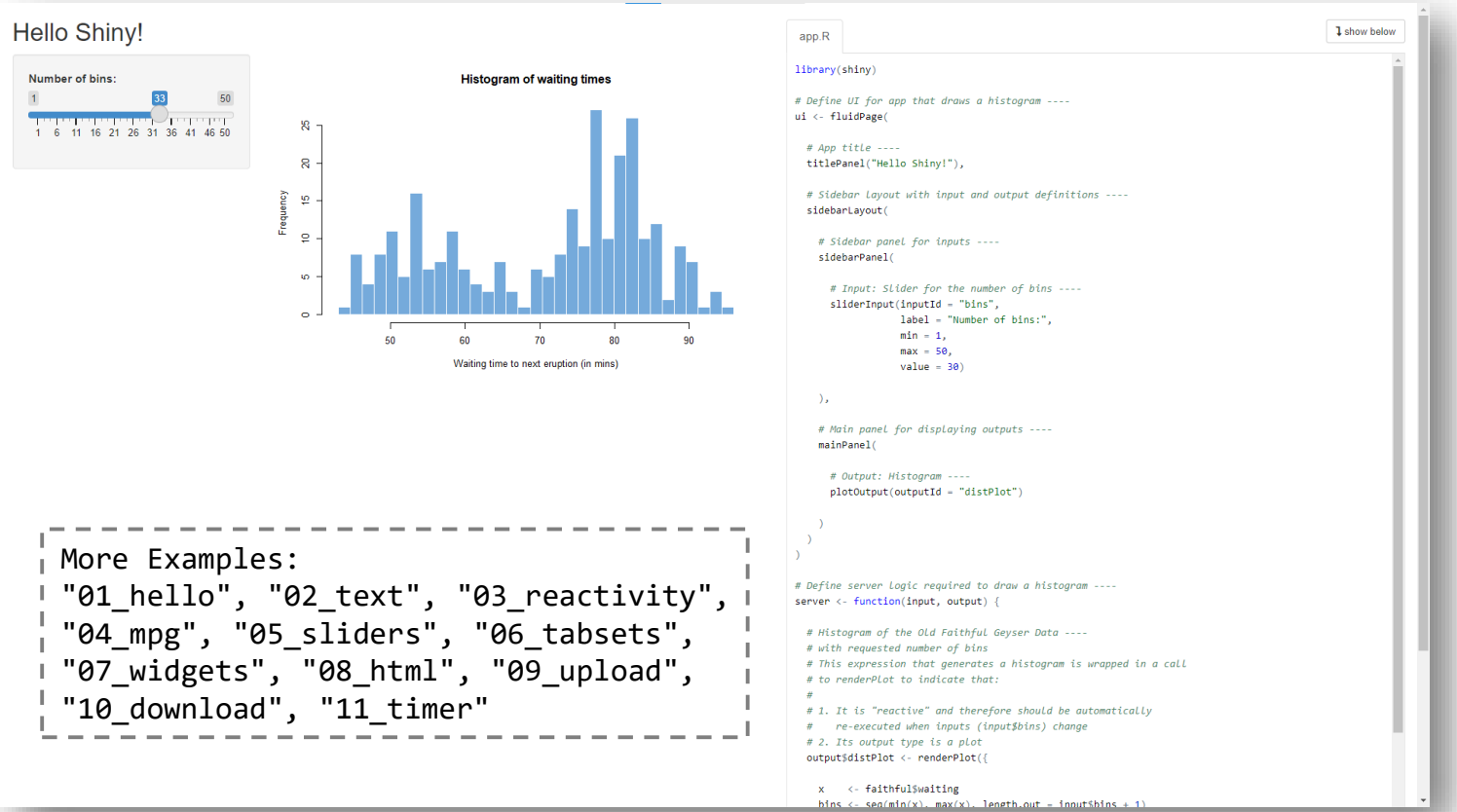
# Interactive Network

https://angustu.shinyapps.io/NetworkApp/



# Running the First R Shiny app

```r
library(shiny)
runExample("01_hello")
```



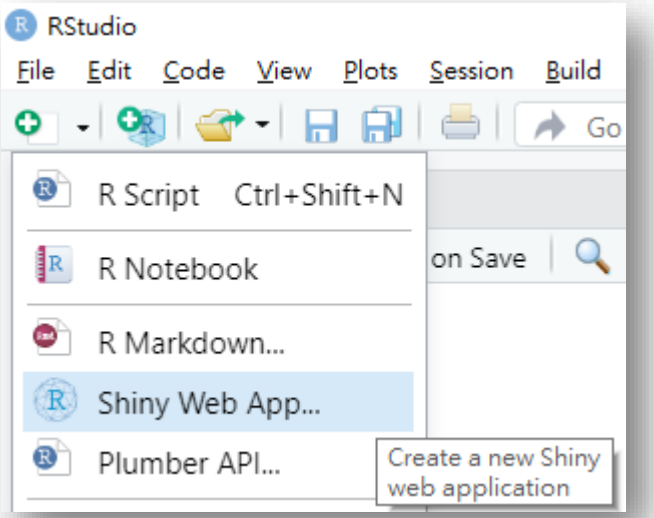

```
More Examples:
"01_hello", "02_text", "03_reactivity",
"04_mpg", "05_sliders", "06_tabsets",
"07_widgets", "08_html", "09_upload",
"10_download", "11_timer"
```

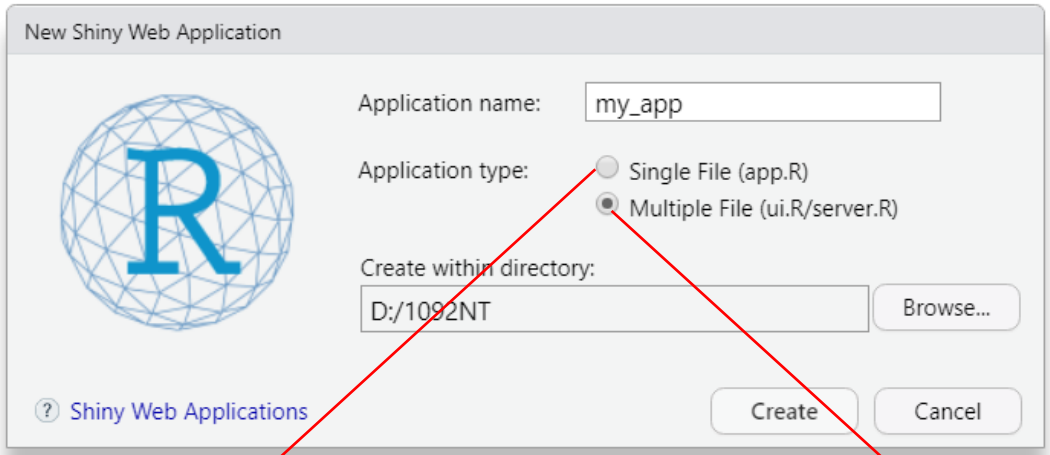
R app.R

```r
library(shiny)


ui = fluidPage(

  titlePanel("Hello Shiny!"),

  sidebarLayout(
     sidebarPanel(sliderInput(……)),
     mainPanel(plotOutput("distPlot"))
  )

)

server = function(input, output) {

    output$distPlot = renderPlot({……})

}

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

Shiny applications have two components, a **user interface object** and a **server function**, that are passed as arguments to the `shinyApp` function that creates a Shiny app object from this UI/server pair.

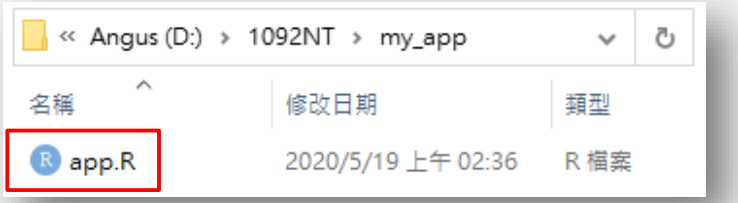
# ShinyApp

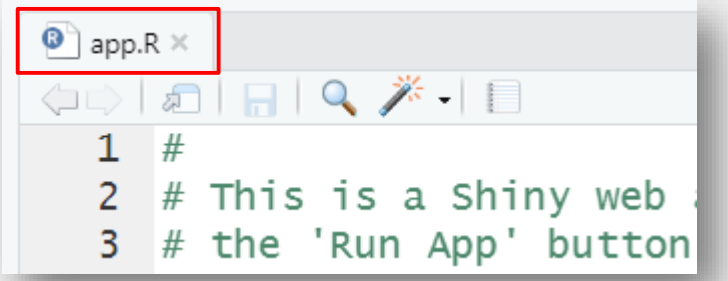


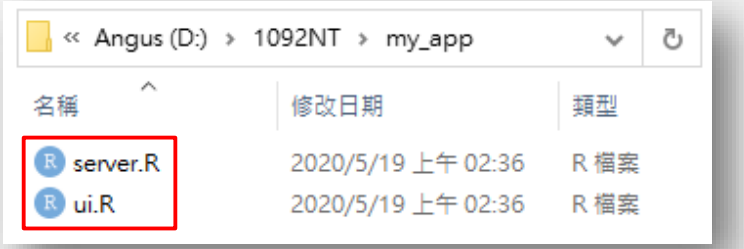

兩者效果相同

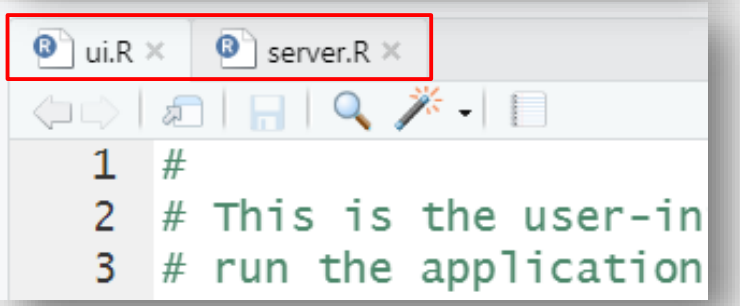
## Single File (app.R)



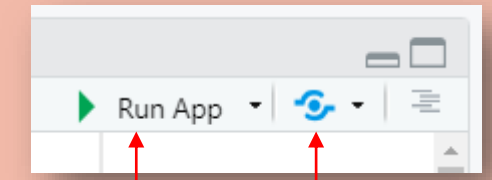




## Multiple File (ui.R/server.R)




## Run ShinyApp




Run App

Publish to web

# ShinyApp

app.R

```
library(shiny)

ui = fluidPage(

  titlePanel("Hello Shiny!"),

  sidebarLayout(
     sidebarPanel(sliderInput(……)),
     mainPanel(plotOutput("distPlot"))
  )

)

server = function(input, output) {

    output$distPlot = renderPlot({……})

}

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

my_app

(以下.R檔要放同個資料夹)

global.R

ui.R

```
library(shiny)

shinyUI(fluidPage(

  titlePanel("Hello Shiny!"),

  sidebarLayout(
     sidebarPanel(sliderInput(……)),
     mainPanel(plotOutput("distPlot"))
  )

))
```

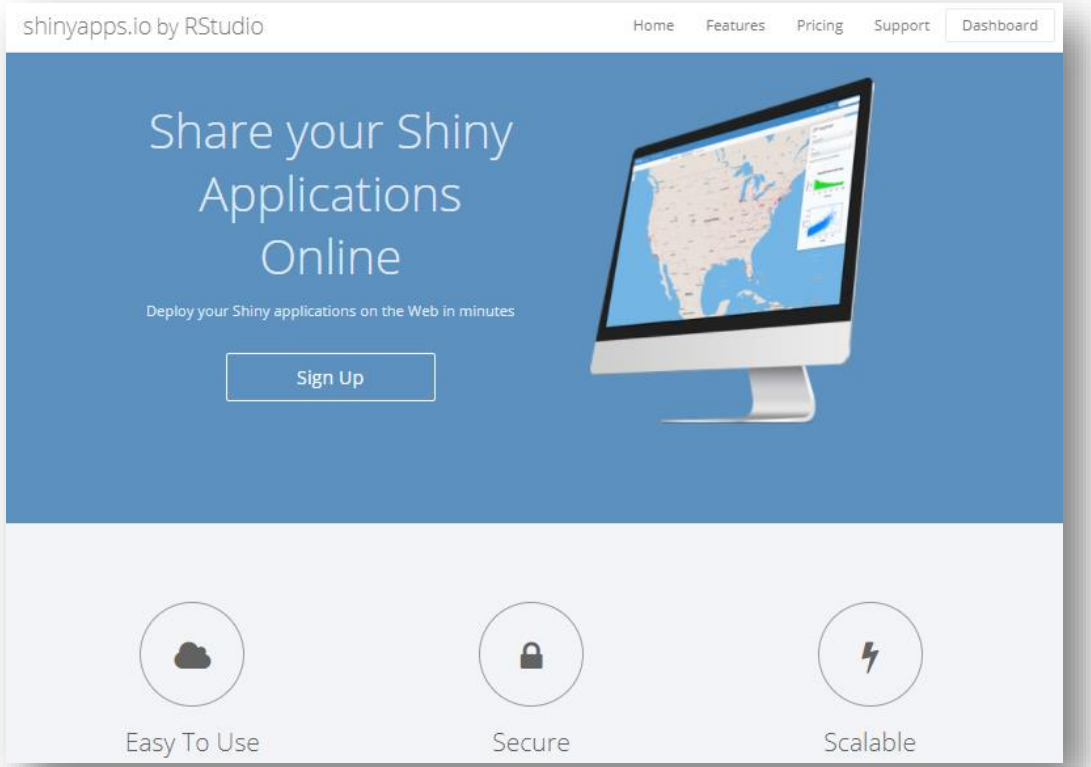
server.R

```
library(shiny)

shinyServer(function(input, output) {

    output$distPlot = renderPlot({……})

})
```

→ 後面會更詳細說明。直接動手操作更好懂！

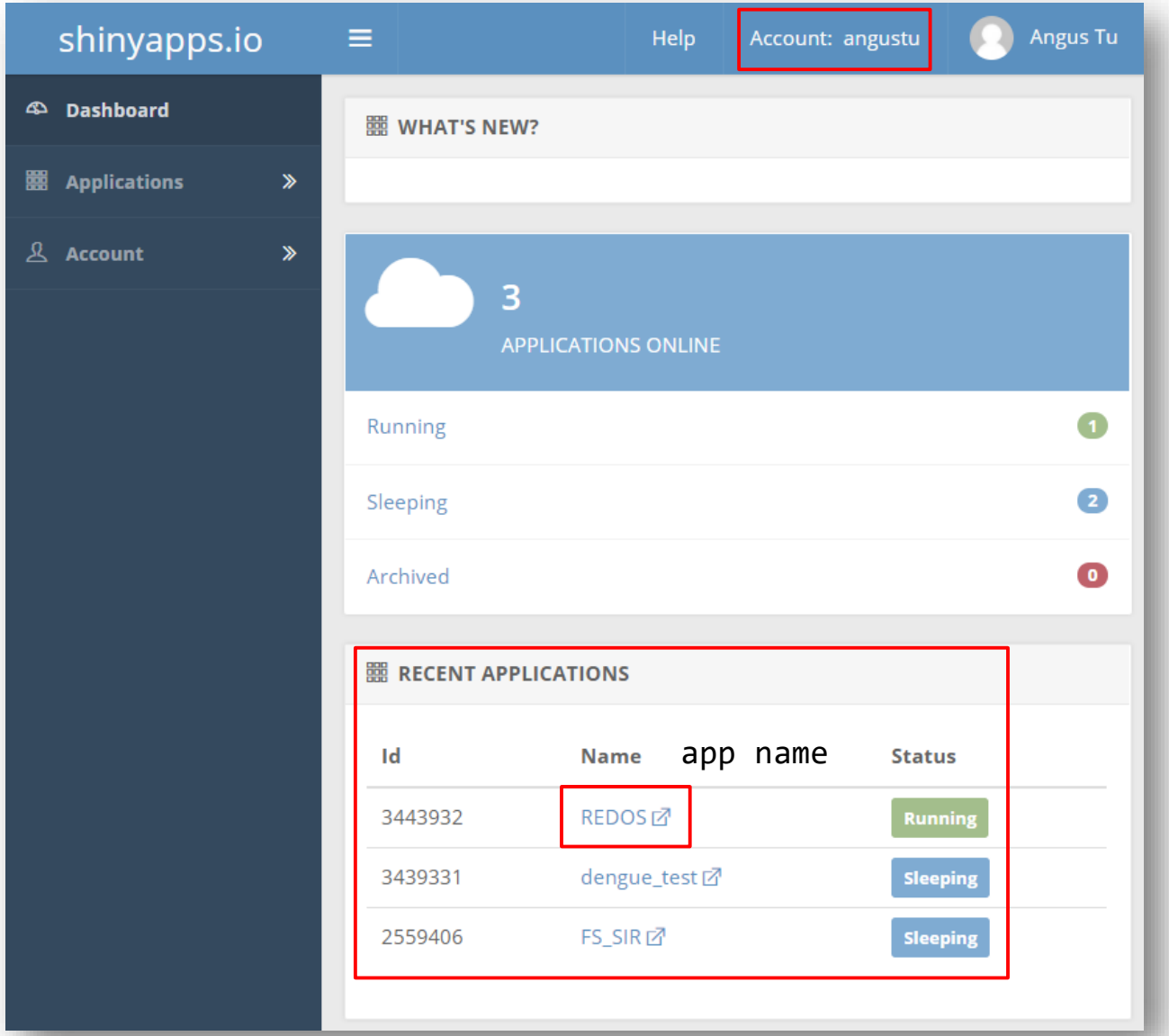
# Shiny App Server

http://www.shinyapps.io/



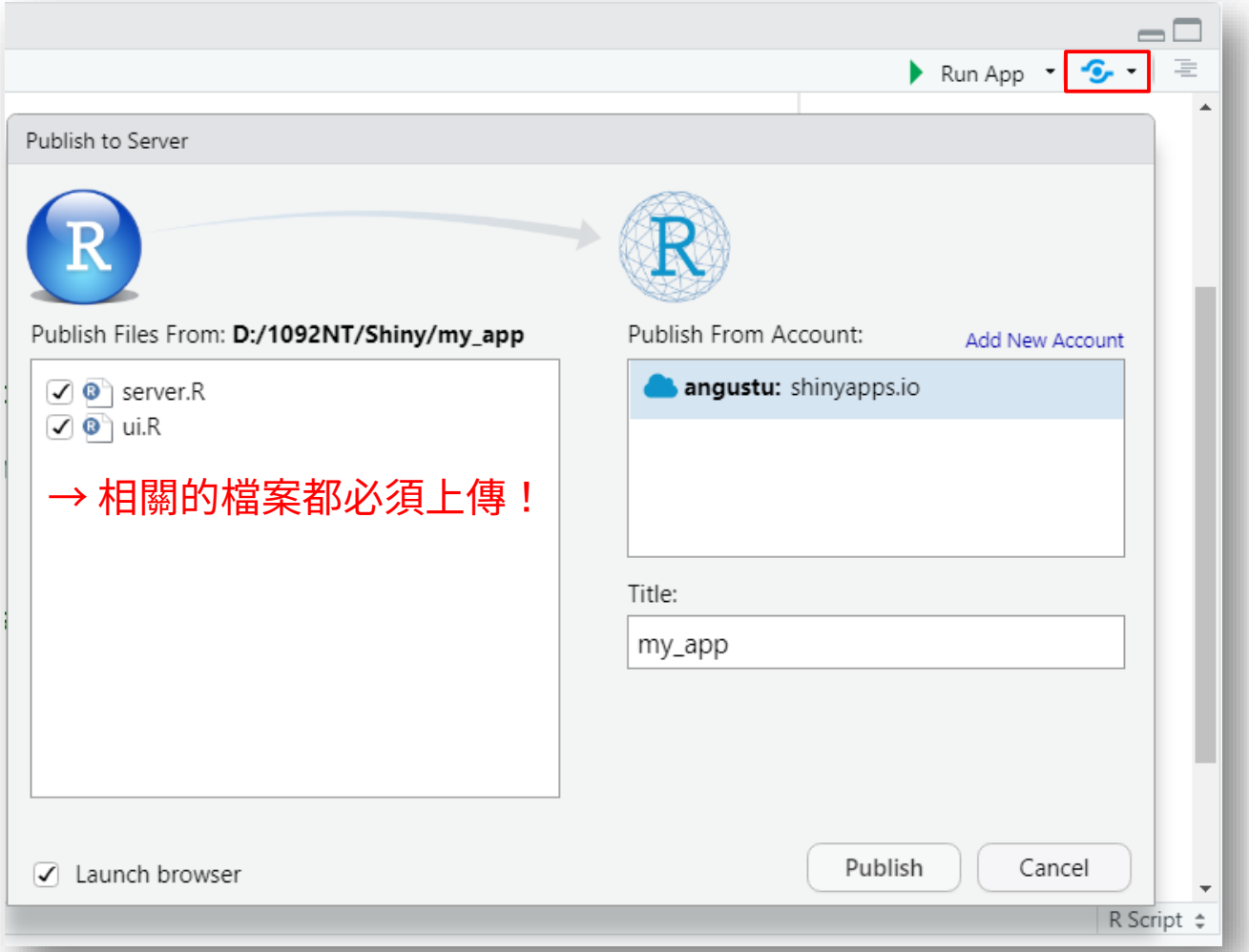

→ Sign Up

※中文問題

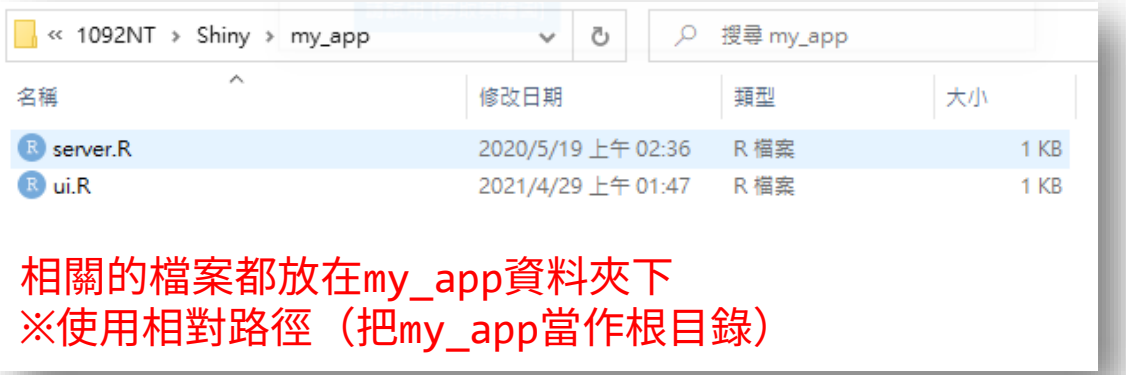


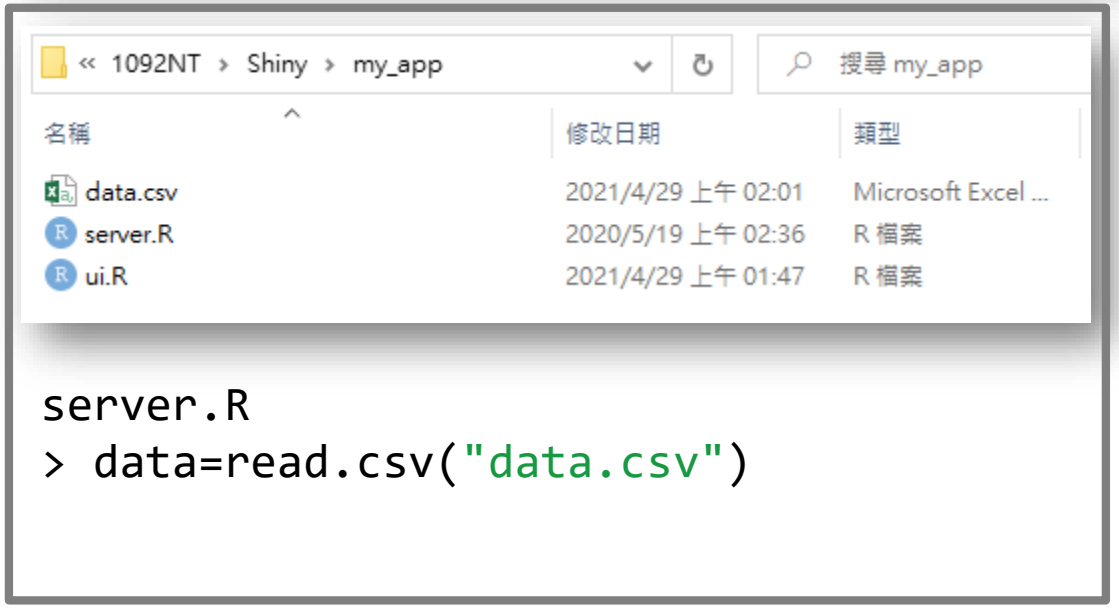

→ https://<account_name>.shinyapps.io/<app_name>/

# Deploying Shiny apps to the web




# Path of Files



# R Shiny Cheatsheet

https://shiny.rstudio.com/images/shiny-cheatsheet.pdf

## Interactive Web Apps with shiny Cheat Sheet

learn more at shiny.rstudio.com

RStudio

### Basics

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).

#### App template

Begin writing a new app with this template. Preview the app by running the code at the R command line.

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

- **ui** - nested R functions that assemble an HTML user interface for your app
- **server** - a function with instructions on how to build and rebuild the R objects displayed in the UI
- **shinyApp** - combines `ui` and `server` into a functioning app. Wrap with **runApp()** if calling from a sourced script or inside a function.

#### Share your app

The easiest way to share your app is to host it on shinyapps.io, a cloud based service from RStudio

1. Create a free or professional account at **http://shinyapps.io**
2. Click the **Publish** icon in the RStudio IDE (>=0.99) or run: **rsconnect::deployApp("**<path to directory>**")**

**Build or purchase your own Shiny Server** at www.rstudio.com/products/shiny-server/

### Building an App - Complete the template by adding arguments to fluidPage() and a body to the server function.

Add inputs to the UI with ***Input()** functions

Add outputs with ***Output()** functions

Tell server how to render outputs with R in the server function. To do this:

1. Refer to outputs with **output$<id>**
2. Refer to inputs with **input$<id>**
3. Wrap code in a **render*()** function before saving to output

```
library(shiny)

ui <- fluidPage(
  numericInput(inputId = "n",
    "Sample size", value = 25),
  plotOutput(outputId = "hist")
)

server <- function(input, output) {
  output$hist <- renderPlot({
    hist(rnorm(input$n))
  })
}

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

Save your template as **app.R**. Alternatively, split your template into two files named **ui.R** and **server.R**.

```
library(shiny)
ui <- fluidPage(
  numericInput(inputId = "n",
    "Sample size", value = 25),
  plotOutput(outputId = "hist")
)

server <- function(input, output) {
  output$hist <- renderPlot({
    hist(rnorm(input$n))
  })
}

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

```
# ui.R
fluidPage(
  numericInput(inputId = "n",
    "Sample size", value = 25),
  plotOutput(outputId = "hist")
)
```

```
# server.R
function(input, output) {
  output$hist <- renderPlot({
    hist(rnorm(input$n))
  })
}
```

**ui.R** contains everything you would save to ui.

**server.R** ends with the function you would save to server.

No need to call **shinyApp()**.

Save each app as a directory that contains an **app.R** file (or a **server.R** file and a **ui.R** file) plus optional extra files.

- app-name ← The directory name is the name of the app
  - app.R
  - global.R ← (optional) defines objects available to both ui.R and server.R
  - DESCRIPTION ← (optional) used in showcase mode
  - README ← (optional) data, scripts, etc.
  - <other files>
  - www ← (optional) directory of files to share with web browsers (images, CSS, .js, etc.) Must be named "**www**"

Launch apps with `runApp(<path to directory>)`

### 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) |
| **renderText**(expr, env, quoted, func) | foo | **textOutput**(outputId, container, inline) |
| **renderUI**(expr, env, quoted, func) | | **uiOutput**(outputId, inline, container, ...) & **htmlOutput**(outputId, inline, container, ...) |

### Inputs - collect values from the user

Access the current value of an input object with **input $<inputId>**. Input values are **reactive**.

| | |
|---|---|
| Action | **actionButton**(inputId, label, icon, ...) |
| Link | **actionLink**(inputId, label, icon, ...) |
| Choice 1, Choice 2, Choice 3 | **checkboxGroupInput**(inputId, label, choices, selected, inline) |
| Check me | **checkboxInput**(inputId, label, value) |
| | **dateInput**(inputId, label, value, min, max, format, startview, weekstart, language) |
| | **dateRangeInput**(inputId, label, start, end, min, max, format, startview, weekstart, language, separator) |
| Choose File | **fileInput**(inputId, label, multiple, accept) |
| 1 | **numericInput**(inputId, label, value, min, max, step) |
| ••••••• | **passwordInput**(inputId, label, value) |
| Choice A, Choice B, Choice C | **radioButtons**(inputId, label, choices, selected, inline) |
| Choice 1, Choice 1, Choice 2 | **selectInput**(inputId, label, choices, selected, multiple, selectize, width, size) (also **selectizeInput()**) |
| 0, 5, 10 | **sliderInput**(inputId, label, min, max, value, step, round, format, locale, ticks, animate, width, sep, pre, post) |
| Apply Changes | **submitButton**(text, icon) (Prevents reactions across entire app) |
| Enter text | **textInput**(inputId, label, value) |



More cheat sheets at http://www.rstudio.com/resources/cheatsheets/

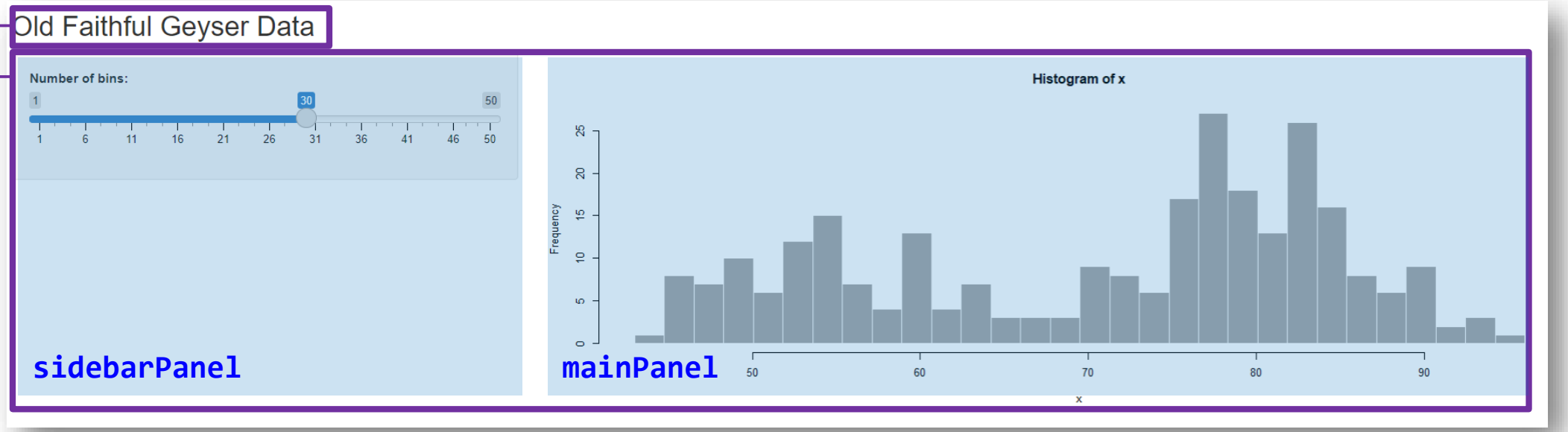
Learn more at **shiny.rstudio.com/tutorial** • shiny 0.12.0 • Updated: 6/15

ui.R

```
library(shiny)

shinyUI(fluidPage(

    titlePanel("Old Faithful Geyser Data"),

    sidebarLayout(
        sidebarPanel(
            sliderInput("bins", "Number of bins:", min = 1, max = 50, value = 30)
        ),

        mainPanel(
            plotOutput("distPlot")
        )
    )
))
```

titlePanel

sidebarPanel



# ui.R ↔ server.R

ui.R ⟷ server.R

Input:

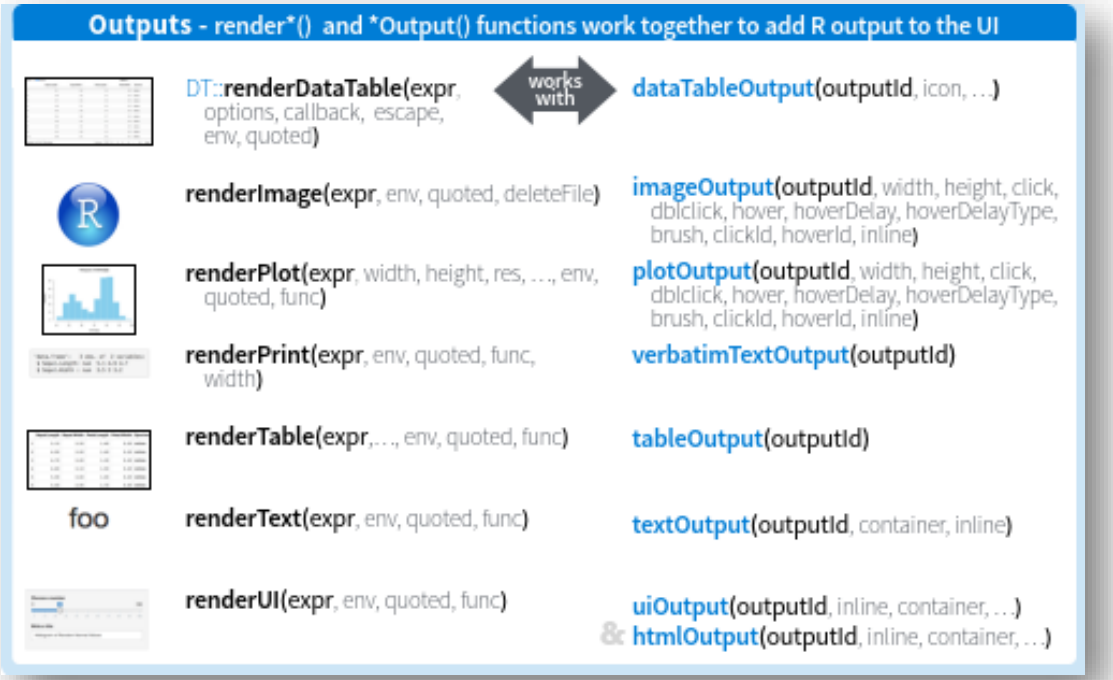
```
sliderInput("bins", "Number of bins:",
min = 1, max = 50, value = 30)
```

Output:

```
plotOutput("distPlot")
```

```
shinyServer(function(input, output) {

    output$distPlot = renderPlot({

        x     = faithful[, 2]
        bins  = seq(min(x),max(x),length.out = input$bins + 1)
        hist(x, breaks = bins, col = 'darkgray')

    })

})
```

input內容要在render區才能互動



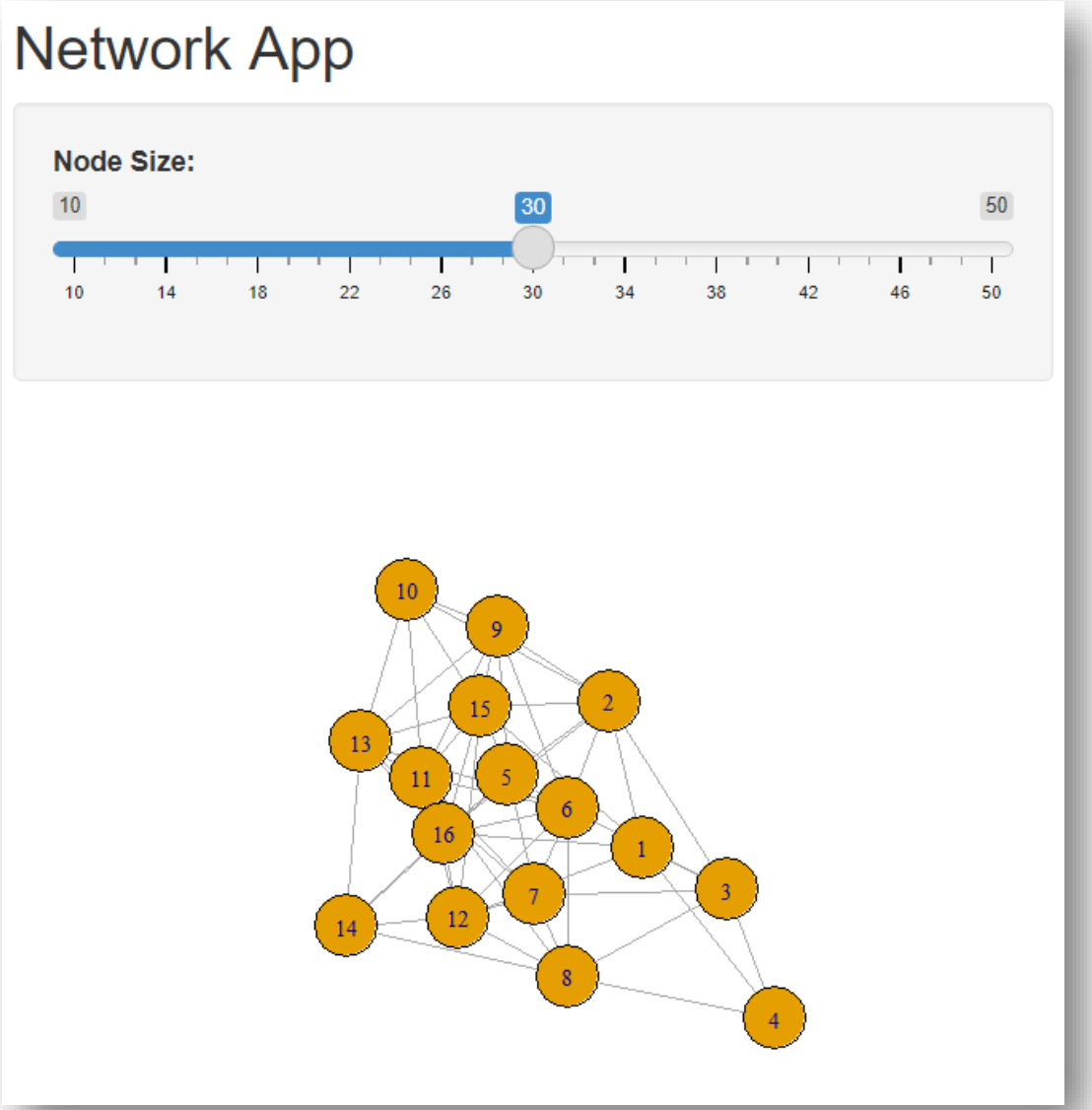
## 實習練習

- 讀取soc-tribes.txt的資料
- sidebar(input)
  - sliderInput: 調整node大小
- main(output)
  - plotOutput: 網絡圖

# 實習參考

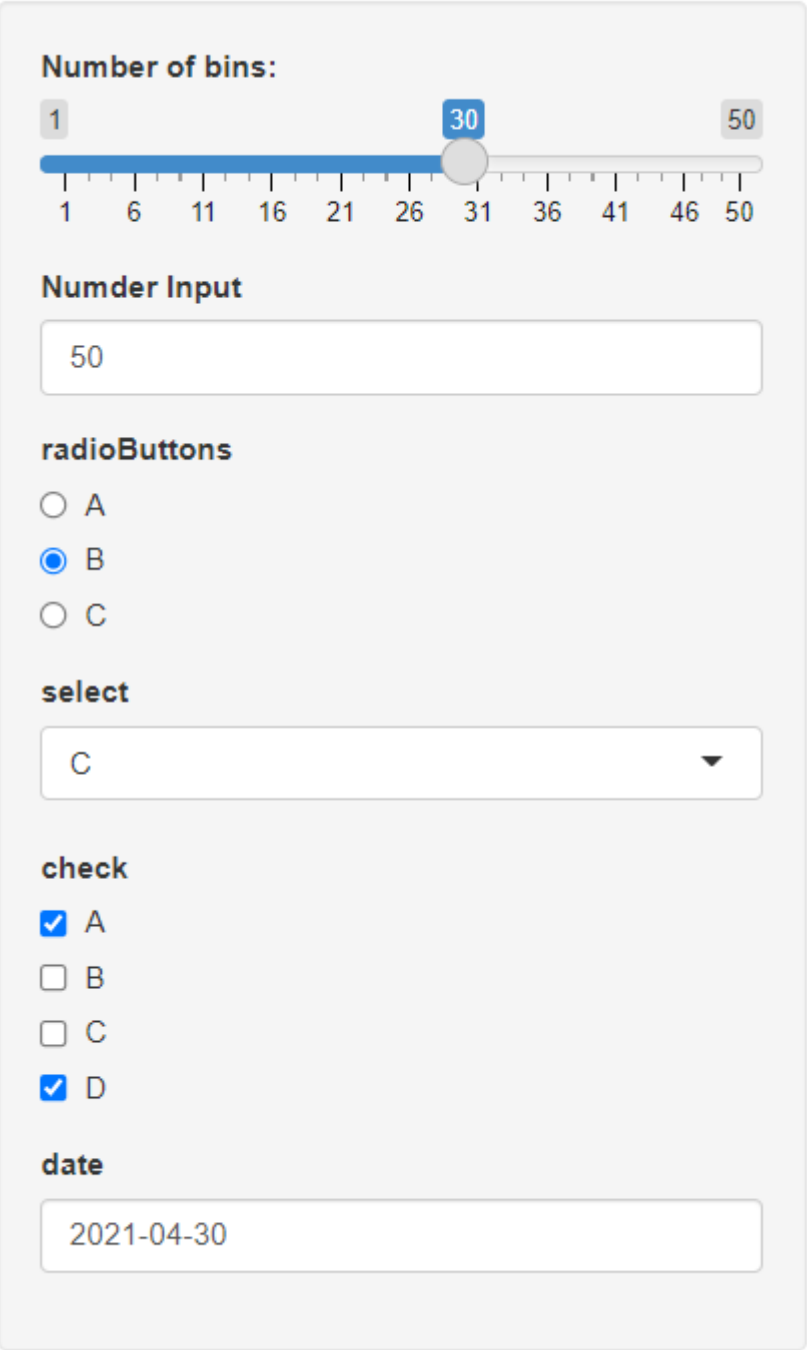
NT_app

```
#ui.R

library(shiny)
shinyUI(fluidPage(
  titlePanel("Network App"),
  sidebarLayout(
    sidebarPanel(
      sliderInput("nodesize","Node Size:",10, 50, 30)
    ),
    mainPanel(
      plotOutput("netPlot")
    )
  )
))

=====================================================
#server.R

library(shiny);library(igraph)
edgelist=read.table("soc-tribes.txt",F," ")
net=graph.data.frame(edgelist,F,1:16)

shinyServer(function(input, output) {
    output$netPlot=renderPlot({
        plot(net,vertex.size=input$nodesize)
    })
})
```



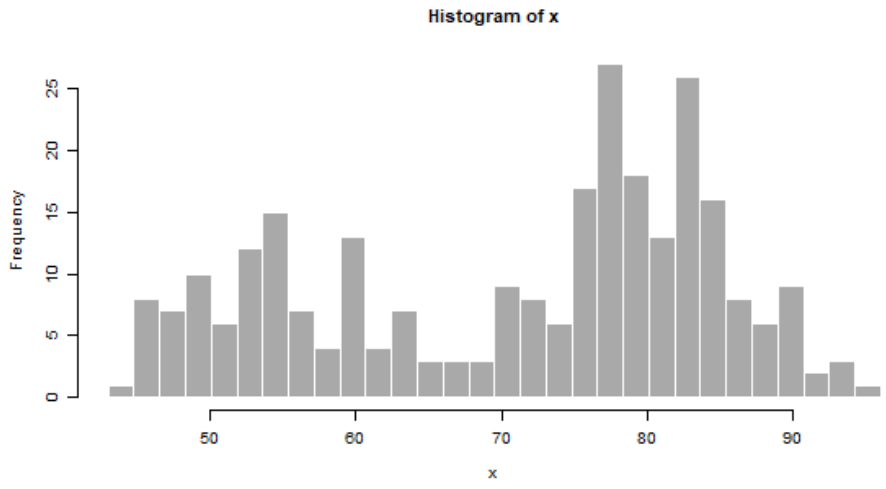
# Input

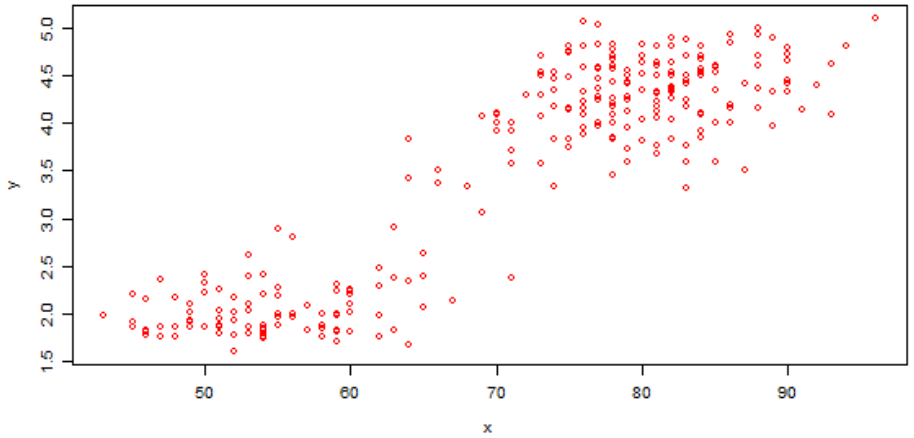
```
sidebarLayout(
    sidebarPanel(
       sliderInput("bins", "Number of bins:", min = 1, max = 50, value = 30),

       numericInput("num","Numder Input",50,1,100,1),

       radioButtons("color","radioButtons",c("A"="black","B"="red","C"="blue"
       ),"red"),

       selectInput("select","select",c("A"="black","B"="red","C"="blue"),"blu
       e"),

       checkboxGroupInput("check","check",c("A","B","C","D"),c("A","D")),

       dateInput("date","date","2021-04-30","2021-01-01","2021-12-31")
    ),

    mainPanel(……)
)
```



# Output

```
mainPanel(
    plotOutput("distPlot"),
    plotOutput("dotPlot")
)
```




```
mainPanel(
    tabsetPanel(
        tabPanel("Histogram", plotOutput("distPlot")),
        tabPanel("Scatter Plot", plotOutput("dotPlot")),
        tabPanel("DataTable",DT::dataTableOutput("dataTable"))
    )
)
```

Histogram | Scatter Plot | DataTable

Show 10 entries  Search:

| | x | y |
|---|---|---|
| 1 | 79 | 3.6 |
| 2 | 54 | 1.8 |
| 3 | 74 | 3.333 |
| 4 | 62 | 2.283 |
| 5 | 85 | 4.533 |
| 6 | 55 | 2.883 |
| 7 | 88 | 4.7 |
| 8 | 85 | 3.6 |
| 9 | 51 | 1.95 |
| 10 | 85 | 4.35 |

Showing 1 to 10 of 272 entries  Previous 1 2 3 4 5 ... 28 Next

## 實習練習

- 讀取soc-tribes.txt的資料

- sidebar(input)
  - sliderInput: 調整node大小
  - sliderInput: 調整edge寬度
  - sliderInput: 調整label大小

- main(output)-tab結構
  - plotOutput: 全網絡圖
  - plotOutput: 個體中心網絡圖
  - dateTableOutput: 每個節點中心性指數