3. 互動式資料視覺化

Interactive Data Visualization

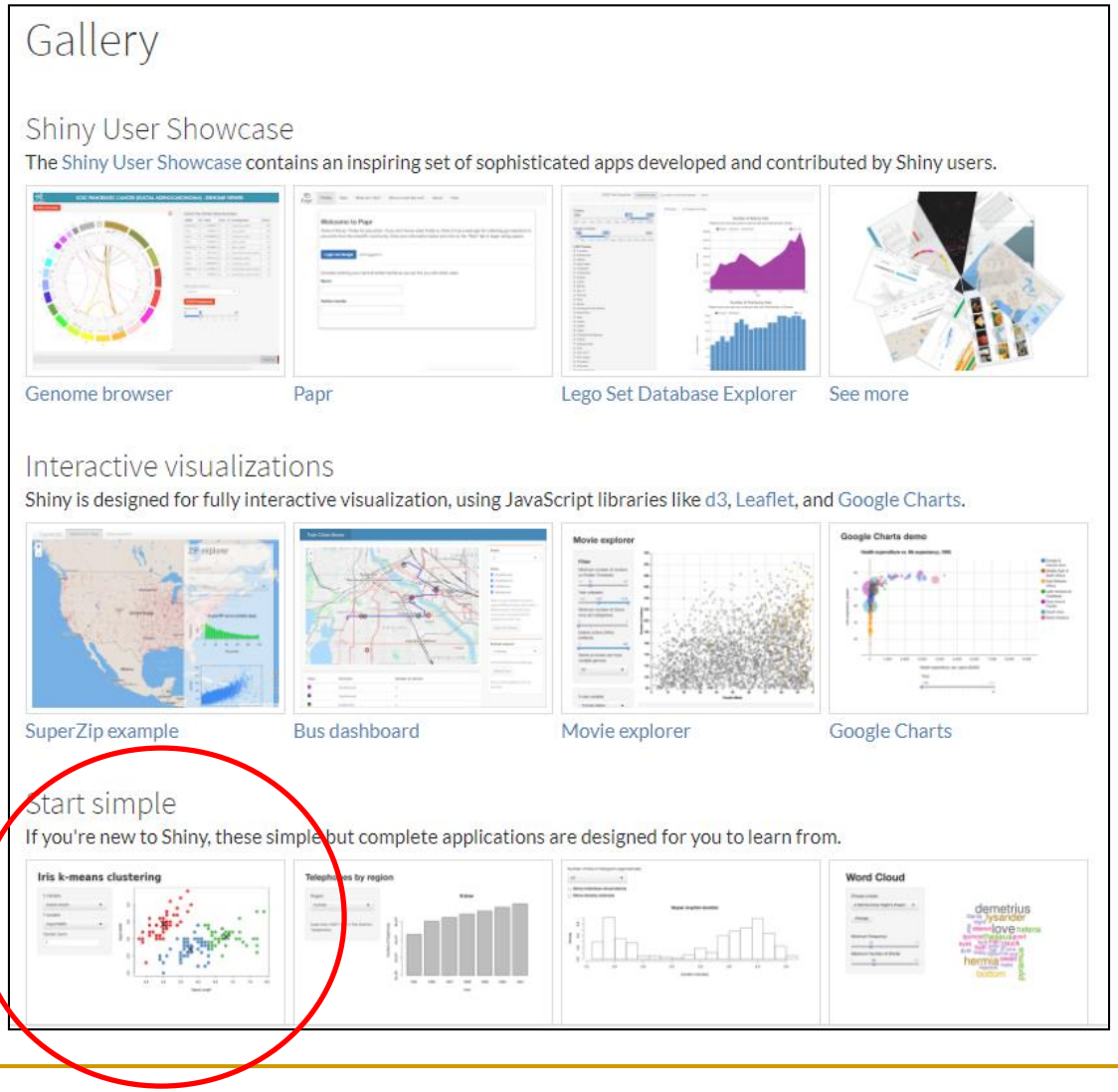
https://ceiba.ntu.edu.tw/1062_Geog5016

授課教師：溫在弘
E-mail: wenthung@ntu.edu.tw

本週課程

- Interactive Data Visualization in R
- Introduction to R Shiny
- Understanding file structure for R Shiny apps
- Building my first R Shiny app
- Deploying Shiny apps to the web

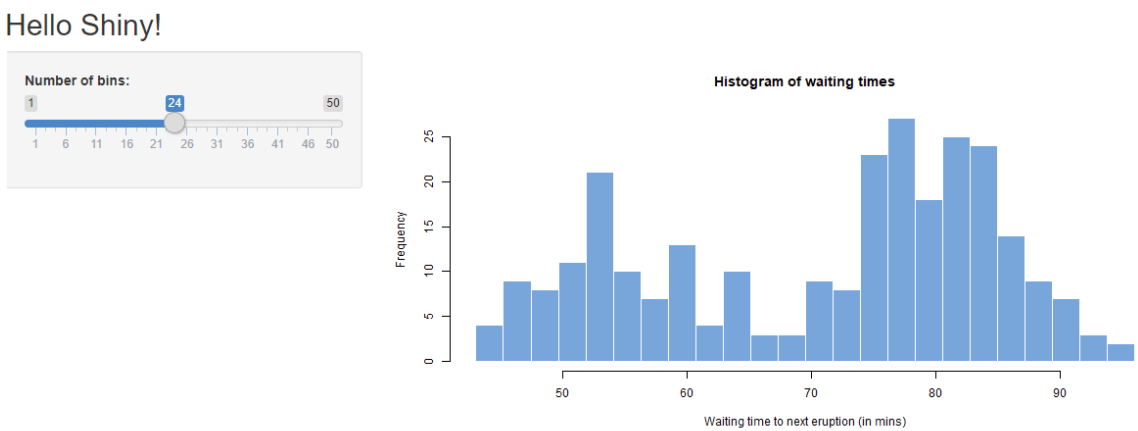
Interactive Data Visualization in R

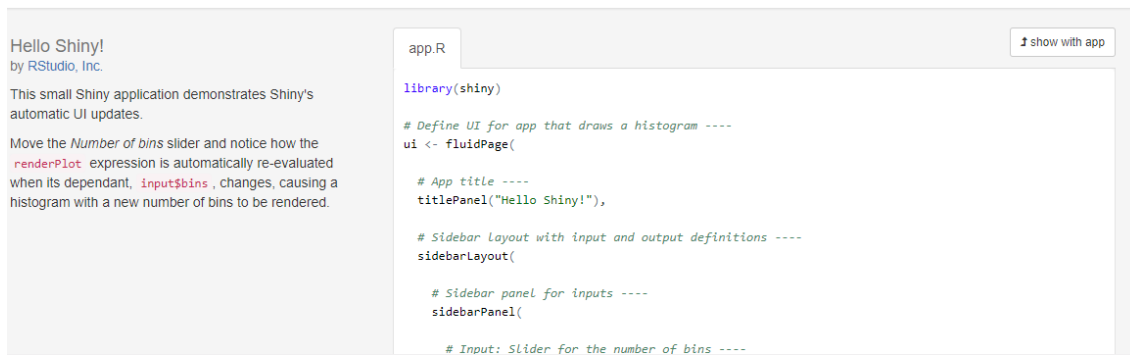


https://shiny.rstudio.com/gallery/

Running the First R Shiny app

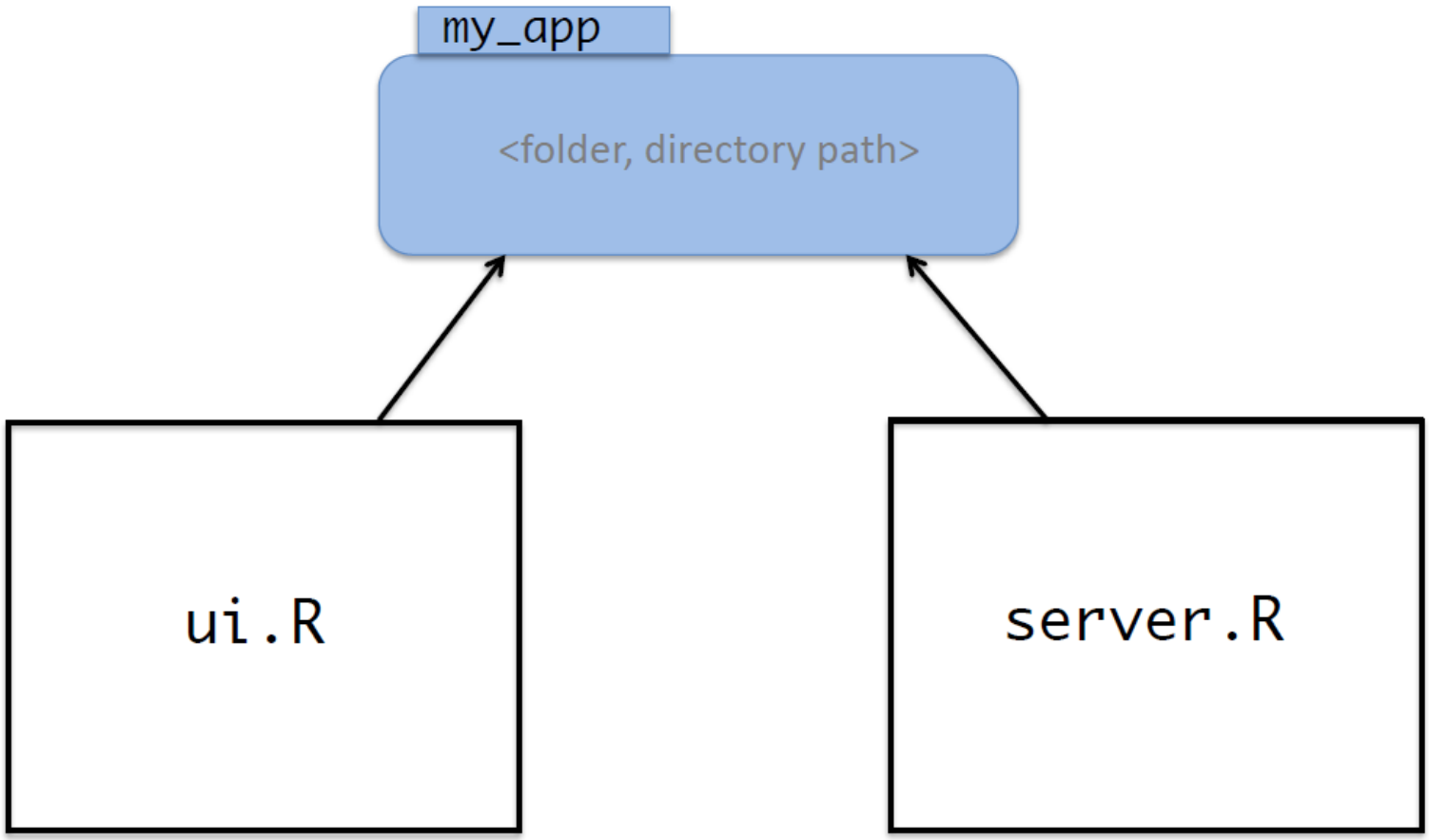
```
library(shiny)

runExample("01_hello")
```





- Shiny applications have two components:
  - **a user-interface definition (UI)** file called `ui.R`
    - This source code is used to set-up what the user will actually see in the web app, i.e. the layout of the web page
      - Title, sliders, widgets, plots, location of items on the page, etc.
    - This source code is also used to accept input from the user
      - e.g. It recognizes what the user has entered in the slider
  - **a server script** file called `server.R`
    - This source code does the computational R work "under the hood" with familiar functions such as *hist(), plot()*, etc.
    - This source code contains the instructions that your computer needs to build your app
- These two source files work together to create your R Shiny web application

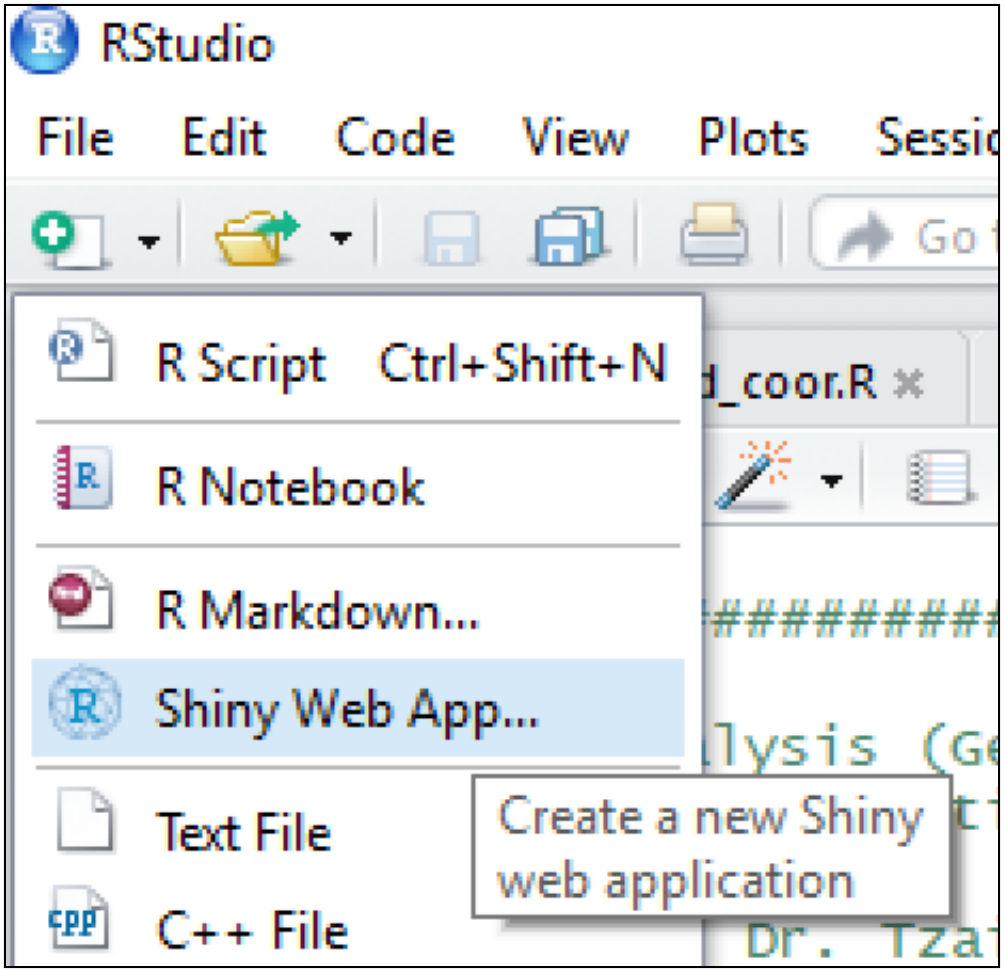
Folder/File structure for R shiny app



To make an R Shiny app, start with this folder/file/filename structure. Put both files (named exactly `ui.R` and `server.R`) into a single folder named for your app.

This is the 'bare-bones' structure for a Shiny app. As you get more complex, you may include other things in this folder, such as a data file, or the 'global.R' file, but that's further down the road.

Creating `ui.R` and `server.R`



Example `ui.r` file from tutorial "Hello Shiny!"

(setting-up the structure of the web page)

ui.R

```
library(shiny)

# Define UI for application that draws a histogram
shinyUI(fluidPage(

  # Application title
  titlePanel("Hello Shiny!"),

  # Sidebar with a slider input for the 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")
    )
  )
))
```

Create a layout with a sidebar (1/3 space of page) and main area (2/3 space of page).

Define sidebar: Put the slider for input in the sidebar panel and name the input as "bins".

Define your slider and set initial settings for slider (value=30).

Define main panel: Put the generated plot in the main panel.

Give your output plot a name, such as "`distPlot`". This name will also be used in the `server.r` file.

Example `server.r` file from tutorial "Hello Shiny!"

(the "under the hood" computations)

server.R

```
library(shiny)

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

  # Expression that generates a histogram. The expression is
  # wrapped in a call to renderPlot to indicate that:
  #
  #  1) It is "reactive" and therefore should re-execute automatically
  #     when inputs change
  #  2) Its output type is a plot

  output$distPlot <- renderPlot({
    x    <- faithful[, 2]  # Old Faithful Geyser data
    bins <- seq(min(x), max(x), length.out = input$bins + 1)

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

Name of output plot stated in the `ui.r` file, or "`distPlot`".

Set-up arguments for the *hist()* function based on user-input "bins" from web app.

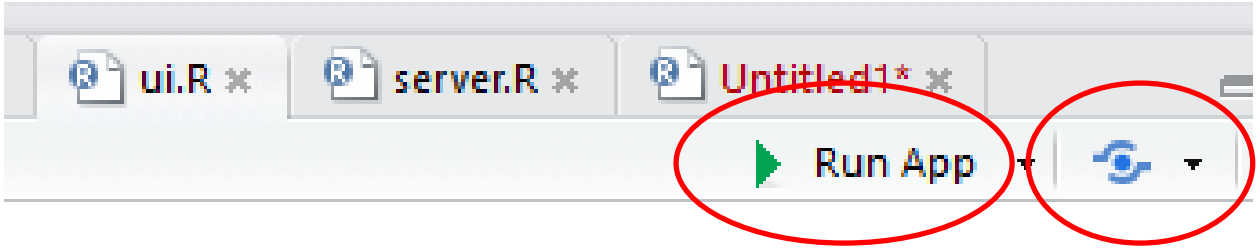
Generate the *hist()* plot with given arguments.

Running an R Shiny App

- Every Shiny app has the same structure:
  - two R scripts saved together in a directory. At a minimum, a Shiny app has ui.R and server.R files.
- You can create a Shiny app by making a new file directory and saving a ui.R and server.R file inside it. **Each app will need its own unique directory (or folder)**.
- You can run a Shiny app by giving the name of its directory to the function *runApp()*.

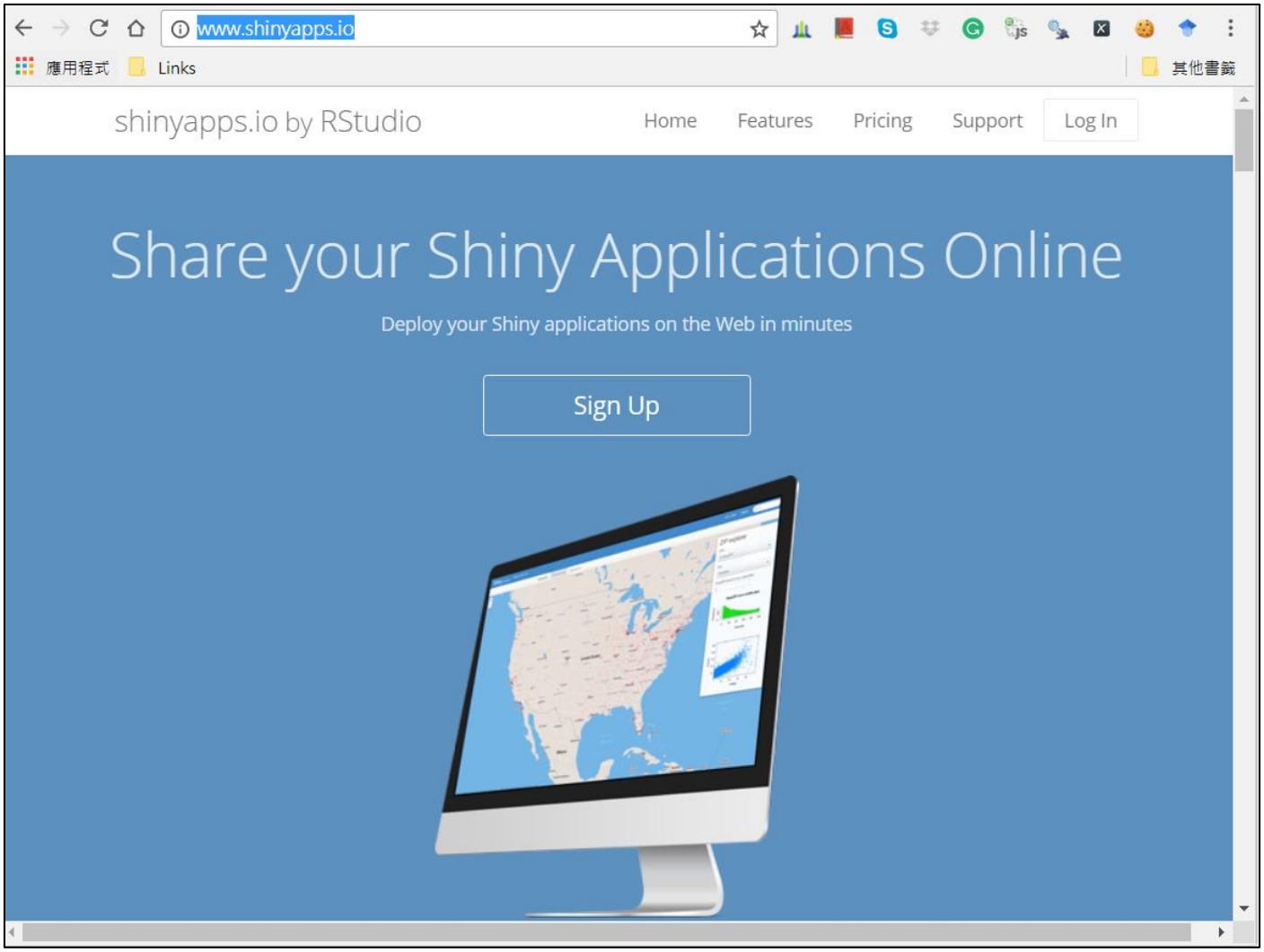
```
> library(shiny)
> runApp("my_app")
```

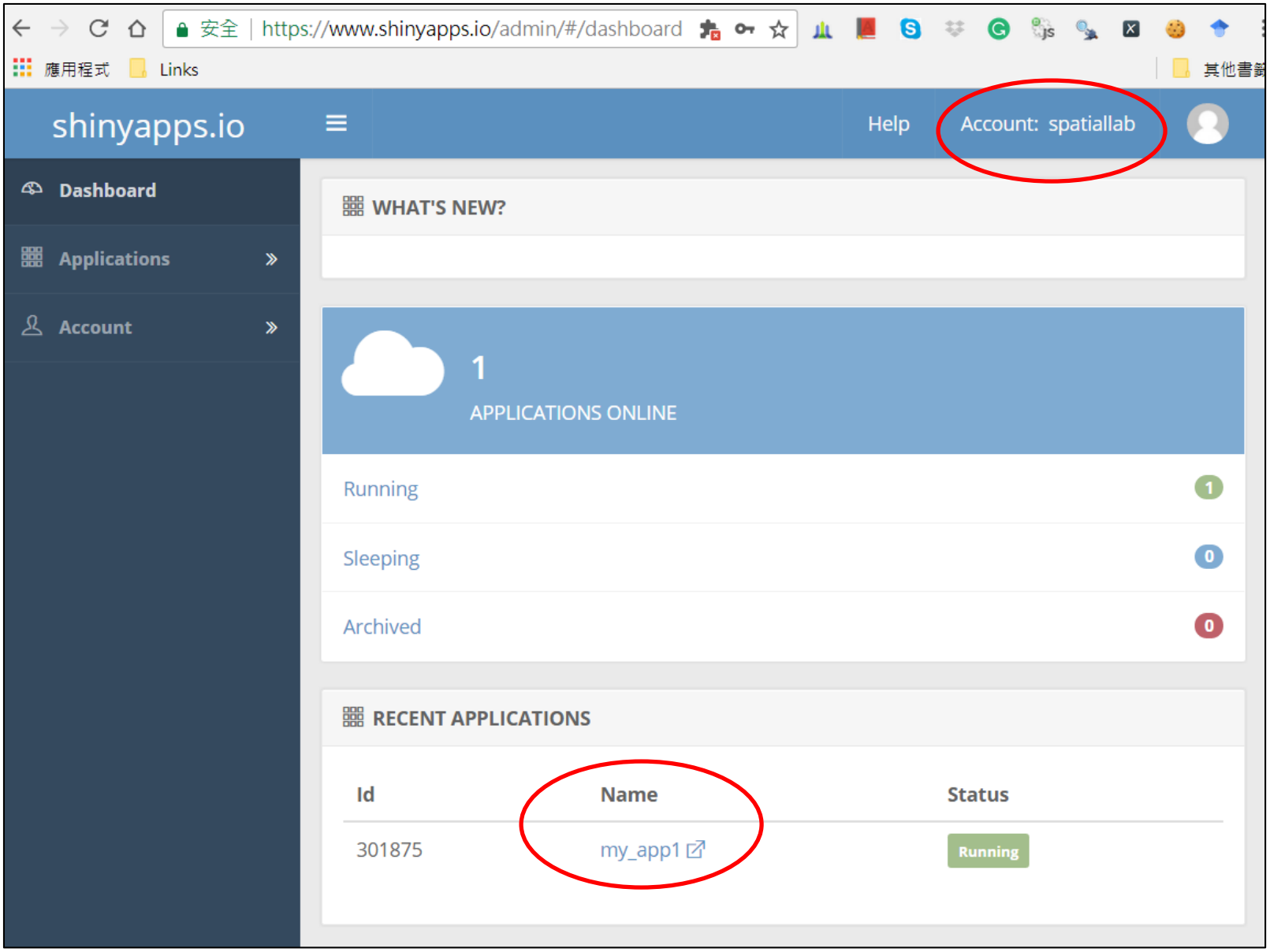
Running the "Hello shiny" app directly from the `ui.R` and `server.R` files



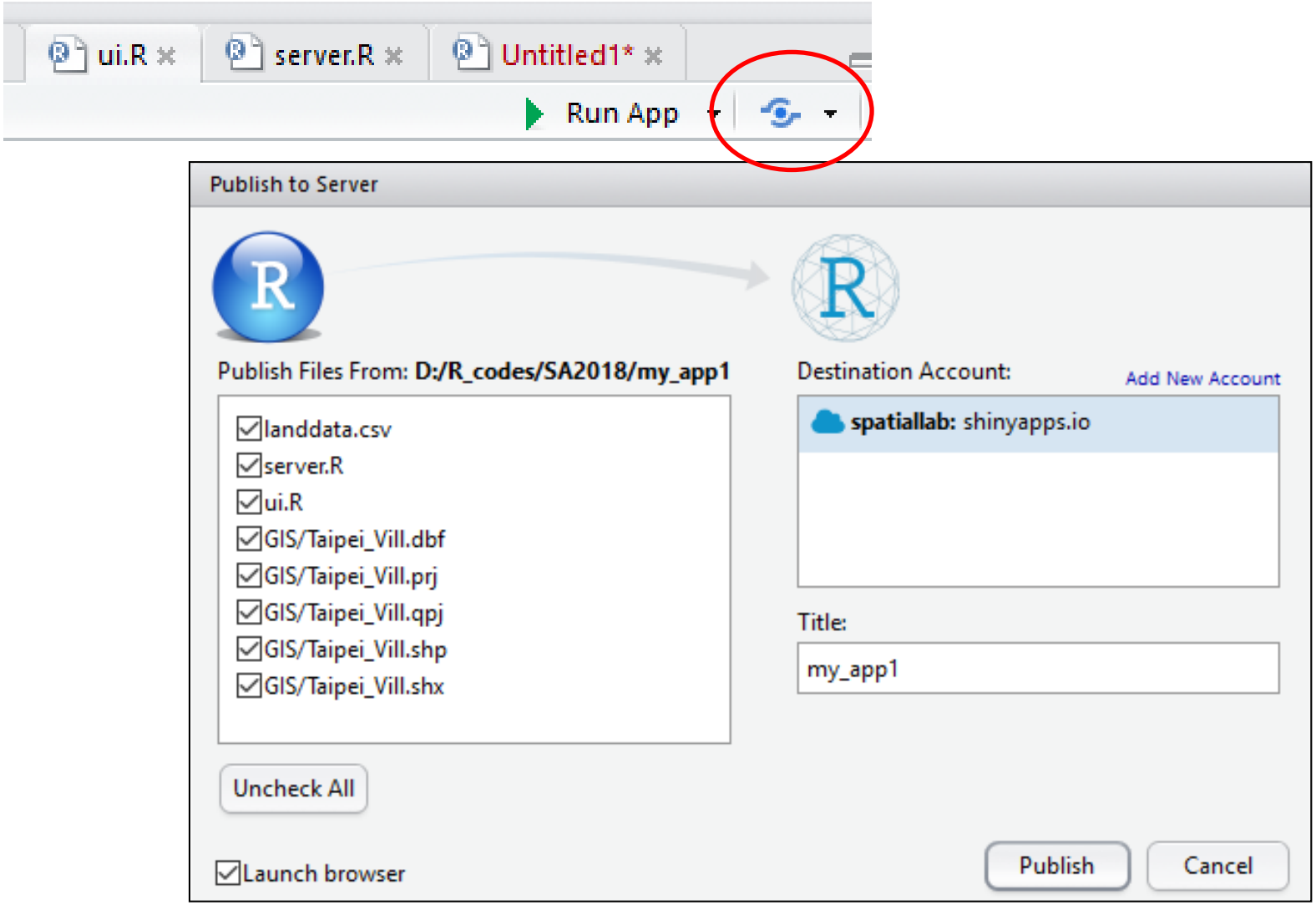
Publish to web

Shiny App Server http://www.shinyapps.io/




安全 | https://www.shinyapps.io/admin/#/dashboard
應用程式
Links
其他書籤
shinyapps.io
Help
Account: spatiallab
Dashboard
Applications
Account
WHAT'S NEW?
1
APPLICATIONS ONLINE
Running
1
Sleeping
0
Archived
0
RECENT APPLICATIONS
Id
Name
Status
301875
my_app1
Running

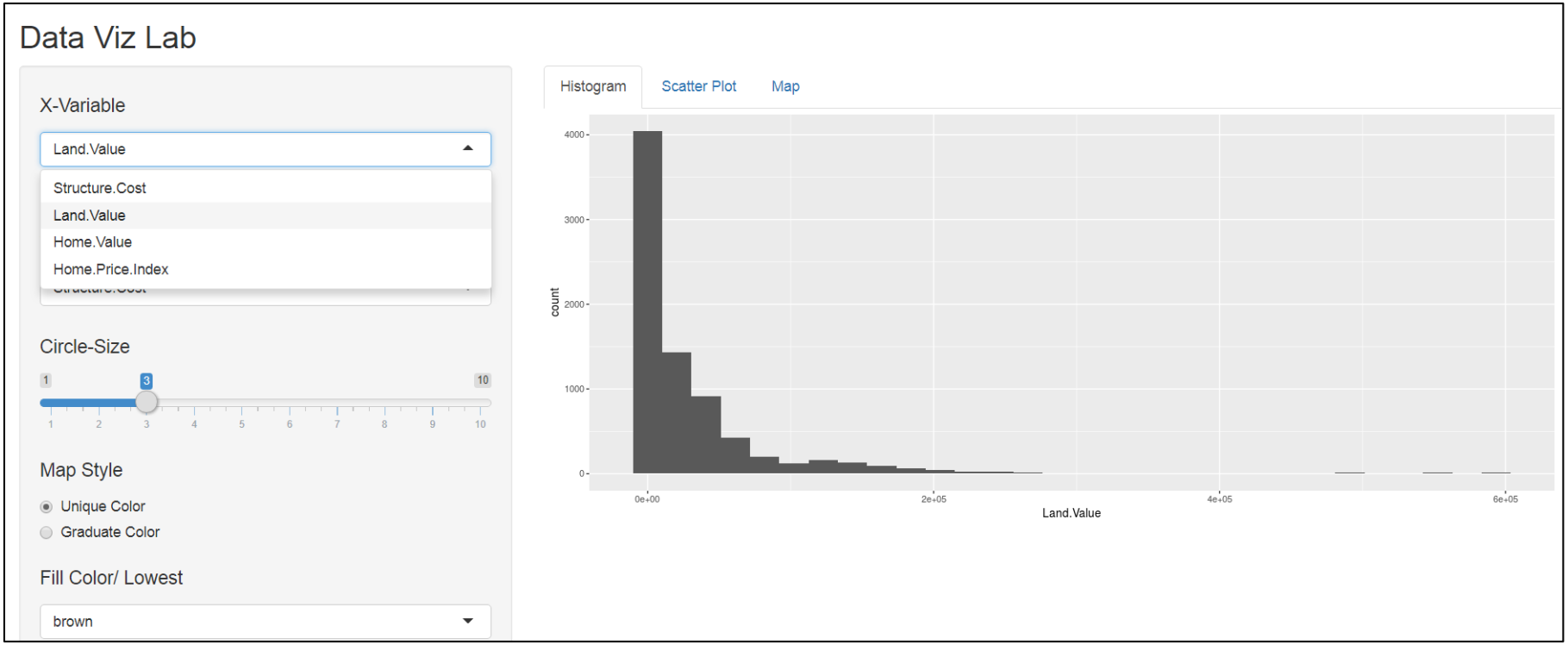
Deploying Shiny apps to the web



https://shiny.rstudio.com/articles/deployment-web.html

範例程式說明

https://spatiallab.shinyapps.io/my_app1/

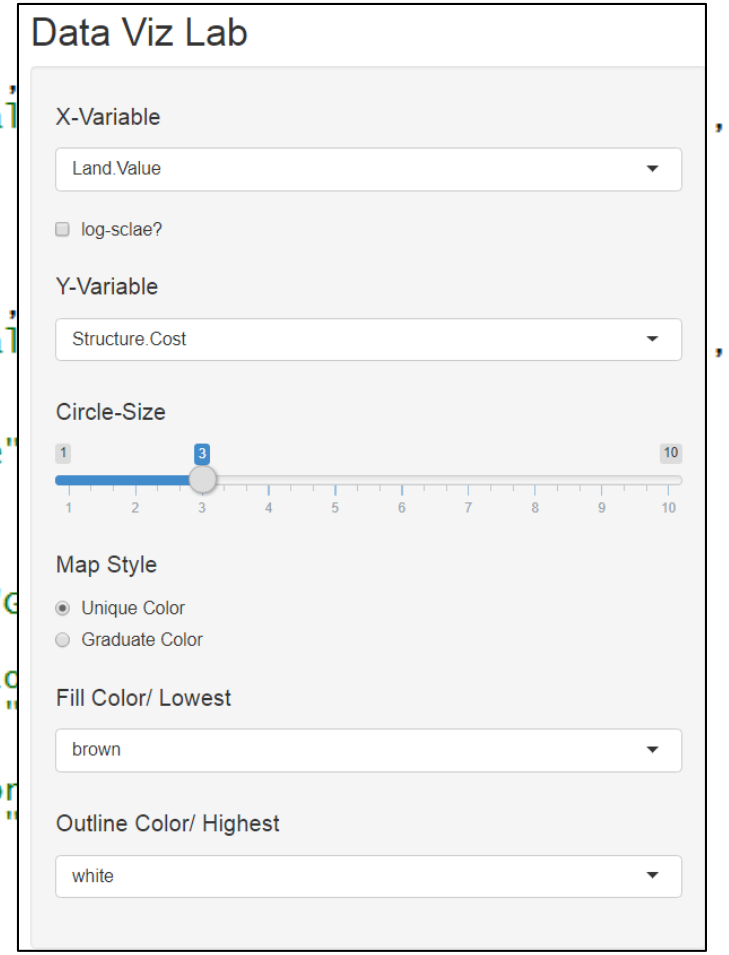


ui.R

```
shinyUI(fluidPage(

  titlePanel("Data Viz Lab"),

  sidebarLayout(

    sidebarPanel(
      selectInput("variable1", label = h4("X-Variable"),
                  choices = c("Structure.Cost","Land.Value","Home.Value","Home.Price.Index"),
                  selected = "Land.Value"),

      checkboxInput("x_checkbox", label = "log-sclae?", value = FALSE),

      selectInput("variable2", label = h4("Y-Variable"),
                  choices = c("Structure.Cost","Land.Value","Home.Value","Home.Price.Index"),
                  selected = "Structure.Cost"),

      sliderInput("circlesize", label = h4("Circle-Size"), min = 1, max = 10, value = 3),


      radioButtons("mapstyle", label = h4("Map Style"),
                   choices = list("Unique Color" = 1, "Graduate Color" = 2), selected = 1),

      selectInput("mapcolor", label = h4("Fill Color/ Lowest"),
                  choices = c("brown","yellow","green","blue","red"), selected = "brown"),

      selectInput("mapcolor2", label = h4("Outline Color/ Highest"),
                  choices = c("white","yellow","green","blue","red"), selected = "white")

    ),

    mainPanel(
      plotOutput("distPlot")
```

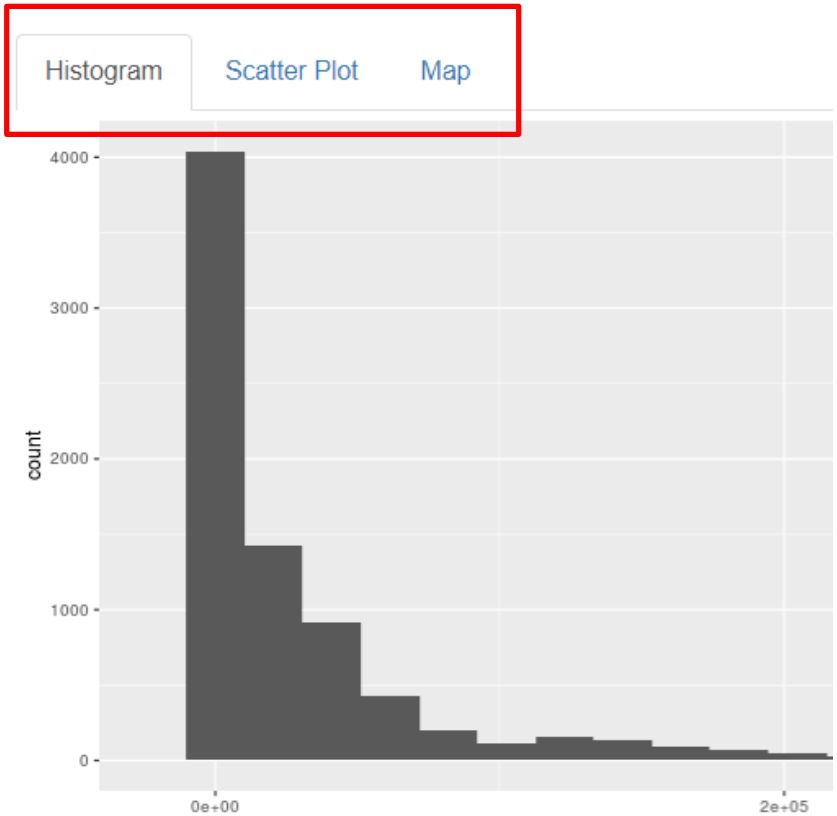
ui.R

```
shinyUI(fluidPage(

  titlePanel("Data Viz Lab"),

  sidebarLayout(

    sidebarPanel(
      selectInput("variable1", label = h4("X-Variable"),
                  choices = c("Structure.Cost","Land.Val
                  selected = "Land.Value"),

      checkboxInput("x_checkbox", label = "log-sclae?",

      selectInput("variable2", label = h4("Y-Variable"),
                  choices = c("Structure.Cost","Land.Val
                  selected = "Structure.Cost"),

      sliderInput("circlesize", label = h4("Circle-Size"


      radioButtons("mapstyle", label = h4("Map Style"),
                   choices = list("Unique Color" = 1, "G

      selectInput("mapcolor", label = h4("Fill Color/ Lo
                  choices = c("brown","yellow","green","

      selectInput("mapcolor2", label = h4("Outline Color
                  choices = c("white","yellow","green","

    ),

    mainPanel(
      plotOutput("distPlot")
```



ui.R

```
mainPanel(
  #plotOutput("distPlot")
  tabsetPanel(
    tabPanel("Histogram", plotOutput("hisPlot")),
    tabPanel("Scatter Plot", plotOutput("distPlot")),
    tabPanel("Map", plotOutput("spatial"))
  )
```



ui.R

```
mainPanel(
  #plotOutput("distPlot")
  tabsetPanel(
    tabPanel("Histogram", plotOutput("hisPlot")),
    tabPanel("Scatter Plot", plotOutput("distPlot")),
    tabPanel("Map", plotOutput("spatial"))
  )
```

server.R

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

  ###
  output$distPlot <- renderPlot({

    if (input$x_checkbox == TRUE ){
      ggplot(hp2001Q1, aes_string(y = input$variable2, x = input$variable1)) +
        geom_point(size=input$circlesize) + scale_x_log10(input$variable1)
    } else {
      ggplot(hp2001Q1, aes_string(y = input$variable2, x = input$variable1)) +
        geom_point(size=input$circlesize)
    }

  })

  ###
  output$hisPlot <- renderPlot({
    ggplot(housing, aes_string(x = input$variable1)) + geom_histogram()
  })
```

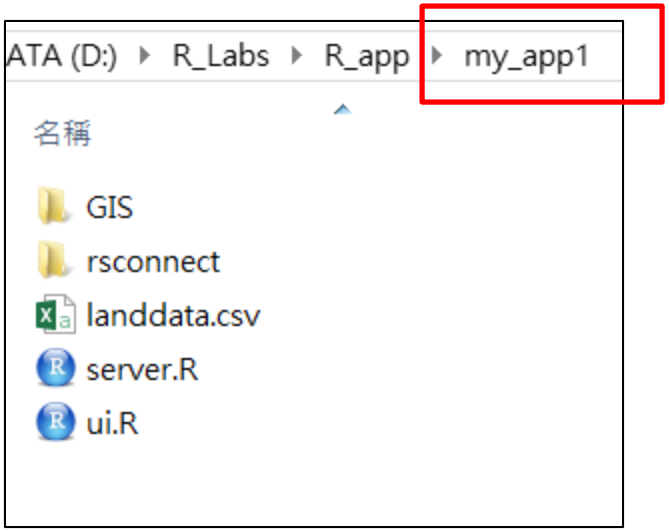
server.R

```
###
output$spatial <- renderPlot({

  map1<- ggplot() +
    geom_polygon(data =Taipei_Vill.f, aes(x=long, y = lat, group = group), fill=input$mapcolor , color=input$mapcolor2) +
    coord_fixed(1.0)

  map2<- ggplot(Taipei_Vill.f, aes_string("long", "lat", group = "group", fill = "density" )) +
         geom_polygon() + coord_equal() + scale_fill_continuous(low = input$mapcolor, high = input$mapcolor2)

  if (input$mapstyle == 1 ){
    map1
  } else {map2}


  })
```

Path of Files

server.R

```
# Data source
housing <- read.csv("landdata.csv")
hp2001Q1 <- subset(housing, Date == 2001.25)

Taipei_Vill <- readOGR(dsn = "GIS", layer = "Taipei_Vill", encoding="utf8")
Taipei_Vill$area<- poly.areas(Taipei_Vill)/10^6
Taipei_Vill$density<- as.numeric(Taipei_Vill$CENSUS) / as.numeric(Taipei_Vill$area)

Taipei_Vill.f <- fortify(Taipei_Vill, region="VILLAGE")
Taipei_Vill.f <- merge(Taipei_Vill.f, Taipei_Vill@data, by.x = "id", by.y = "VILLAGE")
```

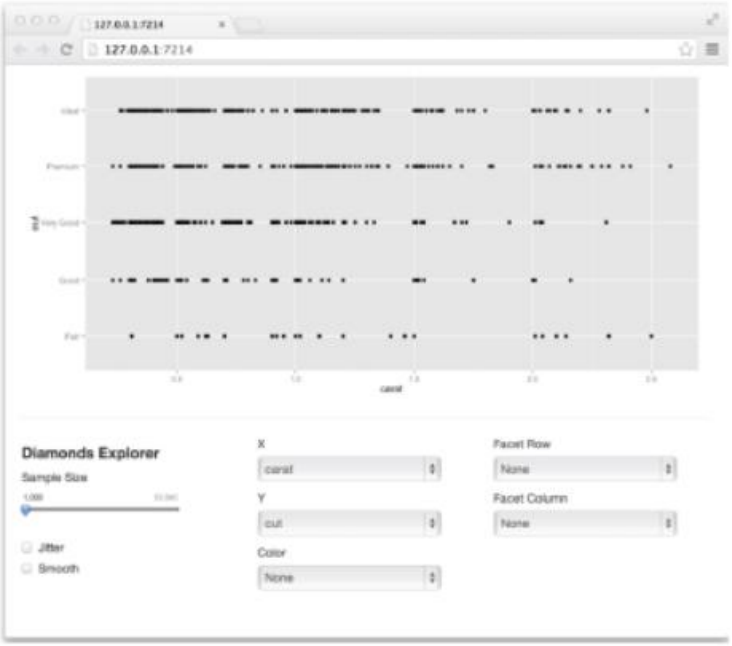


More Examples

R shiny provides 11 specific examples each highlighting a certain ability:

```
> runExample()

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

Different page layout other than sidebar/mainplot setting rows and columns:



See R studio website gallery:
https://shiny.rstudio.com/gallery/

R Shiny 速查表

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

Building an App - Complete the template by adding arguments to fluidPag

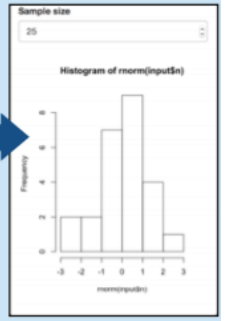
Add inputs to the UI with ***Input()** functions

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
  - app.R — The directory name is the name of the app
  - 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> — (optional) directory of files to share with web browsers (images, CSS, .js, etc.) Must be named "**www**"
  - www

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

實習

- 修改範例程式檔案
  - 在Scatter plot的面板，新增線性迴歸的趨勢線。
  - 在Map的面板，新增 Tpe_Fastfood圖資；並在介面新增 radioButton，可切換檢視

    1.) MIC/KFC 類別、

    2.) Type90 銷售等級 (bubble map)

    3.) Type99 銷售等級 (bubble map)

繳交規定：上傳 pdf 檔，包括 app 網址超連結 以及上述新增功能的截圖畫面

口頭報告的相關時程規劃

- 3/29: consulting TA and Q&A
- 4/05: No class
- 4/12: oral presentation

- **Due: 4/06 (Fri.) 11:59 pm**
  繳交規定：以組為單位，上傳 pdf 檔，包括：app 網址超連結 以及1-page的資料視覺化說明。以應用程式與1-page書面說明評分。(總成績 10%)
- **4/09 公告入選 前10 組名單，準備 ppt 簡報檔。**
- **4/12上課前，須完成上傳 ppt。**
  口頭發表資料視覺化的網路應用程式 。
  (10 min，含現場系統展示)

口頭報告的獎勵方式

- 擬邀請系上老師與研究生，與助教共同評分。
- 第 1 名：期中考 +90分 或 作業總成績 A+
- 第2-3名：期中考 +60分
- 第3-6名：期中考 +40分
- 第7-9名：期中考 +20分
- 第 10名：期中考 +10分

- 修課同學票選 前3名：期中考 +20分