

shiny for Python Tutorial

大數據分析

- R/Python/Julia/SQL 程式設計與應用
(R/Python/Julia/SQL Programming and Application)
- 資料視覺化 (Data Visualization)
- 機器學習 (Machine Learning)
- 統計品管 (Statistical Quality Control)
- 最佳化 (Optimization)



李明昌博士

alan9956@gmail.com

<http://rwepa.blogspot.com/>



大綱

1. RWEPA 簡介
2. 資料分析架構→APC方法
3. 資料分析與視覺化應用
4. shiny 簡介
5. shiny for Python 實作篇1 - my_app
6. shiny for Python 實作篇2 – 01_hello
7. 結論



1.RWEPA 簡介



RWEPA簡介 <http://rwepa.blogspot.com/>

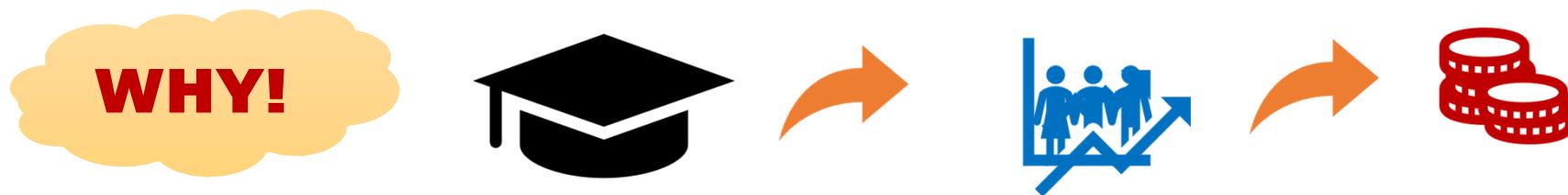
- 姓名：李明昌 (ALAN LEE)
- 現職：中華R軟體學會 常務理事
臺灣資料科學與商業應用協會 常務理事
- 學歷：中原大學 工業與系統工程所 博士
- 經歷：
 - 淡江大學 兼任教師
 - 育達科技大學 兼任教師
 - 佛光大學 兼任教師
 - 國立台北商業大學 兼任教師
 - 東吳大學 兼任教師
 - 育達科技大學 資訊管理系(所) 專任助理教授
 - 崇友實業 行銷企劃專員
 - 國航船務代理股份有限公司 海運市場運籌管理員
- 大專院校、資策會、工業技術研究院、國家發展委員會、中央氣象局、公平交易委員會、各縣市政府與日本名古屋產業大學等公營單位演講達300餘場，2900小時以上。
- 連絡資訊：alan9956@gmail.com



- iPAS 巨量資料分析師 證照推廣
- iPAS 營運智慧分析師 證照推廣

如何學習 shiny & Python &?

- 熟悉教材內容
- 將教材內容的資料集改為工作資料集
- 遇到問題時, 想辦法尋找答案
- 掌握 APC方法
- 掌握 摘要, 繪圖, 建模
- 參考網路應用文章 (進階) & 學術論文





Python 程式設計-李明昌 免費電子書

- <http://rwepa.blogspot.com/2020/02/pythonprogramminglee.html>

主題: Python 程式設計-李明昌 - ipynb

檔名: Python_Programming_Lee_ipynb.zip

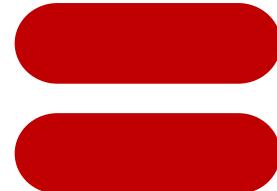
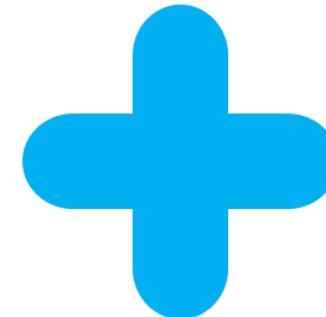
包括 Python 程式設計-李明昌電子書的原始 ipynb 檔案, 圖檔, 部分資料集

下載: https://github.com/rwepa/DataDemo/blob/master/Python_Programming_Lee_ipynb.zip



Python_Programming_Lee_ipynb.zip > python.book.lee >	
名稱	類型
.ipynb_checkpoints	檔案資料夾
data	檔案資料夾
img	檔案資料夾
Python程式設計-李明昌.ipynb	IPYNB 檔案

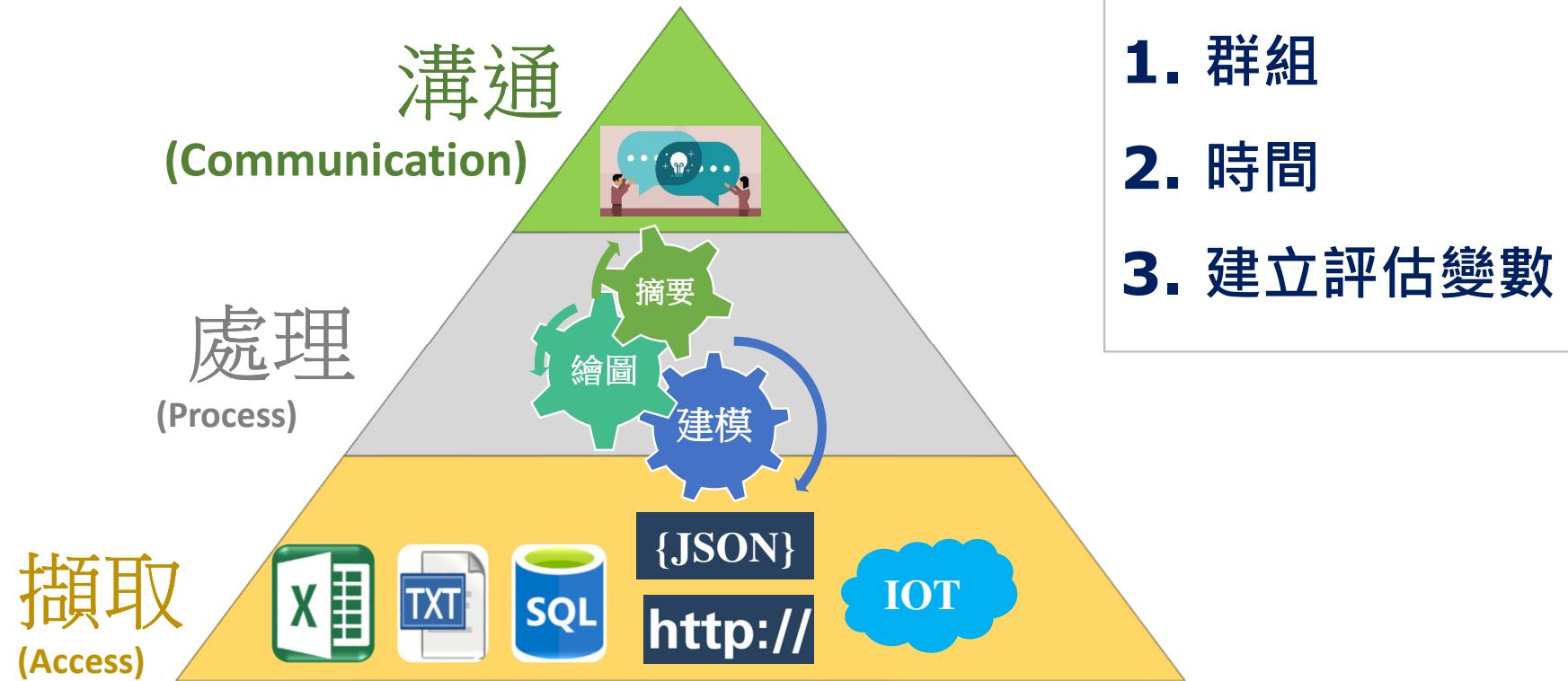
學習目標





2. 資料分析架構→APC方法

★★★資料分析架構→APC方法



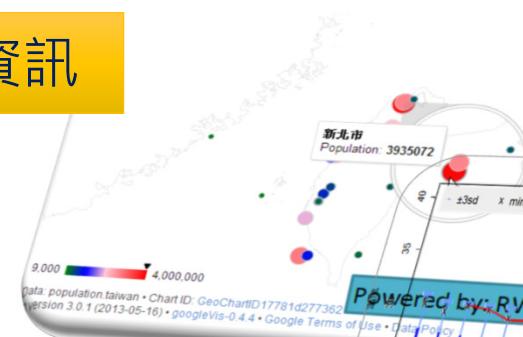


3. 資料分析與視覺化應用

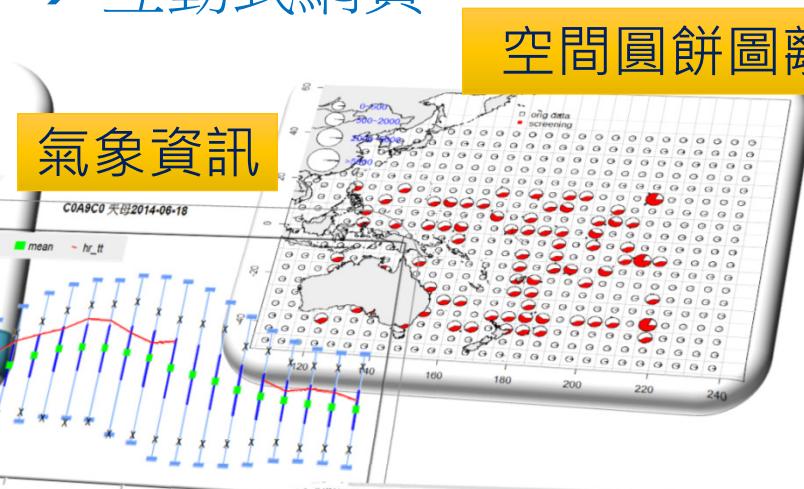
資料分析與視覺化應用

R + shiny → 互動式網頁

地理資訊



氣象資訊



空間圓餅圖離群值分析

保險預測

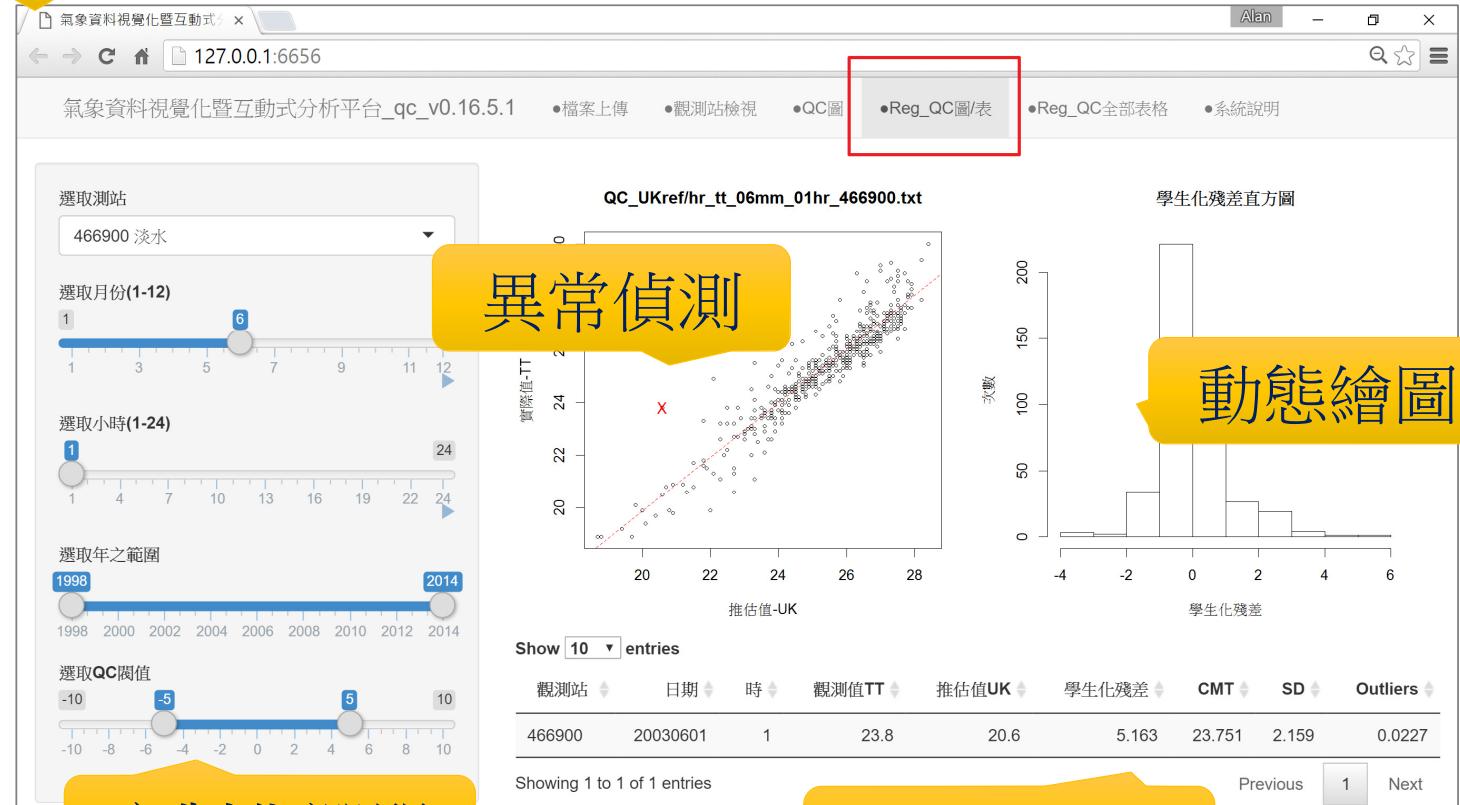


顧客連結資訊



中央氣象局 1,600萬筆資料

網頁呈現



客製化選單

R統計運算

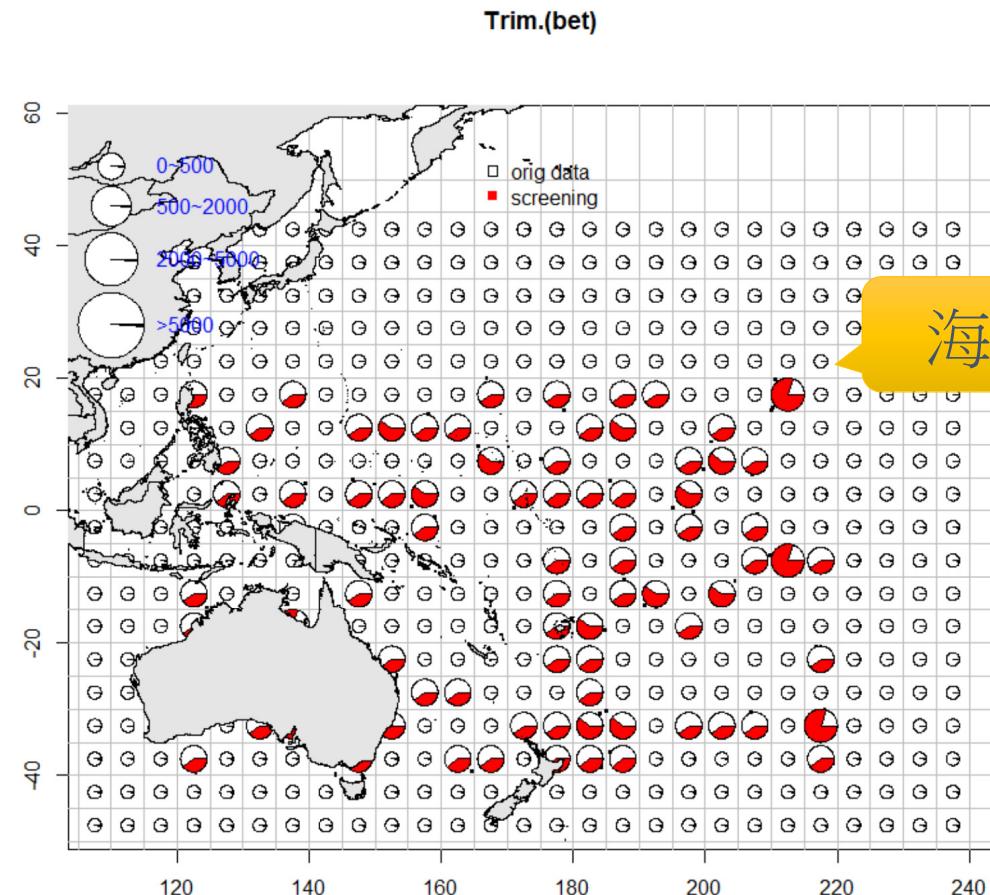
保險預測模型

機率模型閾值調整

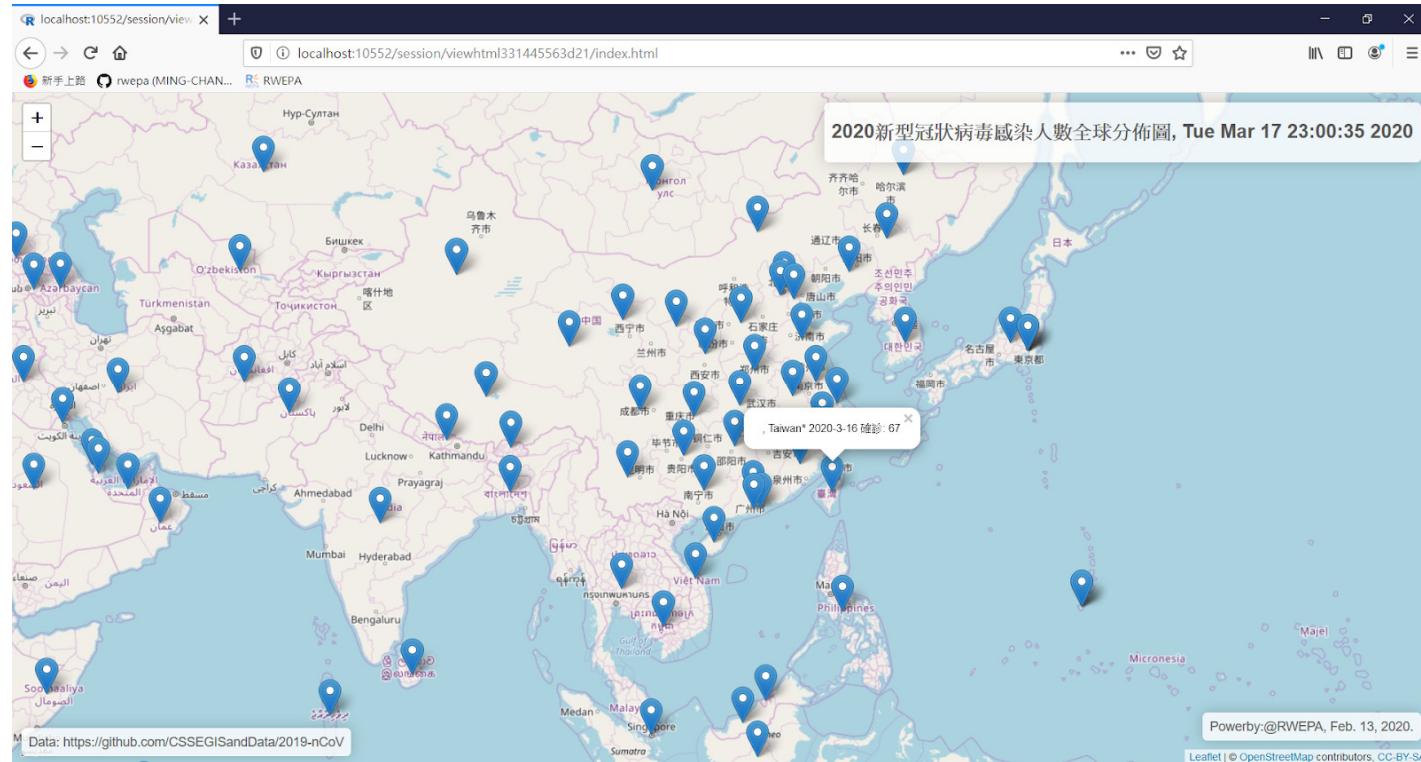
預測結果

性別	女性	車輛種類	私家車	曝露風險		曝露風險對數		無索償折扣	被保險人年齡	私家車 一車齡 0	私家車 一車齡 1	私家車 一車齡 2	私家車 -車齡 0_1_2 組合	車齡 組合	車齡 0_1_2	預測機率	理賠
				曝露風險	曝露風險對數	私家車 一車齡 0	私家車 一車齡 1										
M	0	A	1	0.9144422	-0.08944106	50	4	1	0	0	1	0	2	0.1069	有		
M	0	A	1	0.8158795	-0.20348856	20	4	0	0	1	1	2	2	0.1441	有		
3	M	0	A	1	0.8377823	-0.17699695	50	3	0	0	1	1	2	2	0.1866	有	
4	M	0	A	1	0.4325804	-0.83798702	50	6	0	1	0	1	1	2	0.0944	無	
5	M	0	A	1	0.7173169	-0.33223755	50	4	0	0	1	1	2	2	0.1218	有	
6	M	0	A	1	0.8377823	-0.17699695	50	4	0	0	1	1	2	2	0.1495	有	
7	M	0	A	1	0.8487337	-0.16400975	50	5	0	0	1	1	2	2	0.1422	有	
8	F	1	A	1	0.8268309	-0.19015503	10	3	0	0	1	1	2	2	0.1733	有	
9	M	0	A	1	0.7145791	-0.33606164	0	5	1	0	0	1	0	2	0.0694	無	
10	M	0	A	1	0.3340178	-1.09656101	0	3	0	0	1	1	2	2	0.0783	無	

空間圓餅圖離群值分析



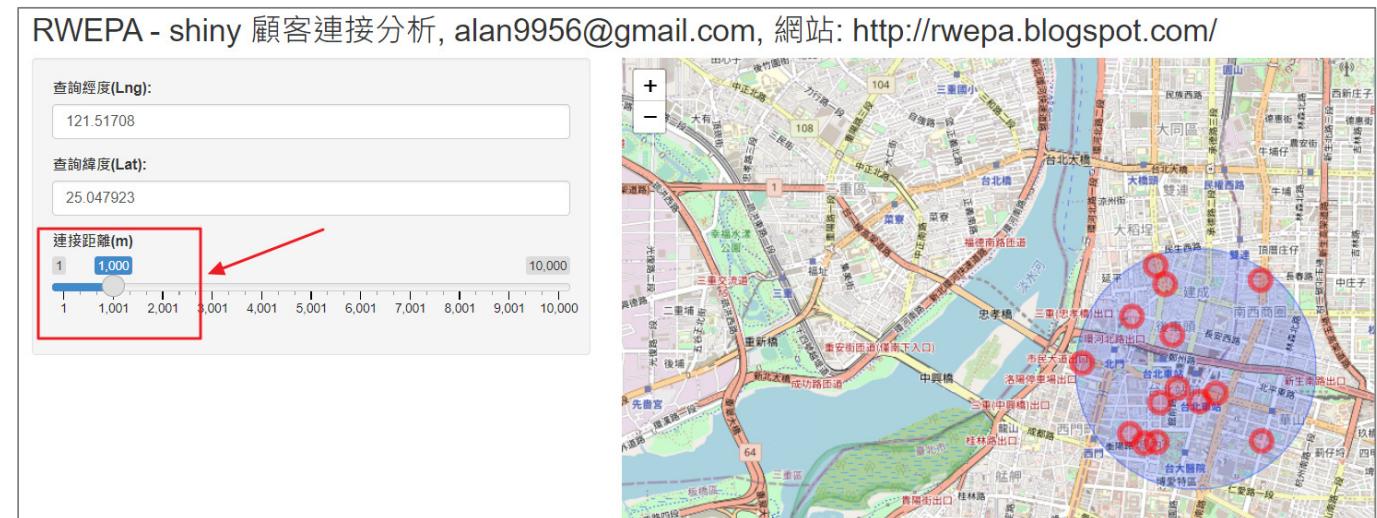
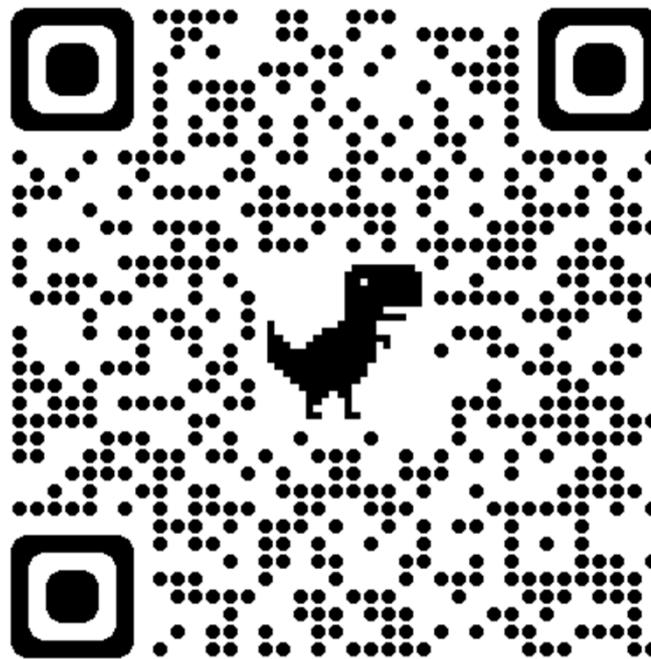
2020新型冠狀病毒視覺化



<http://rwepa.blogspot.com/2020/02/2019nCoV.html>

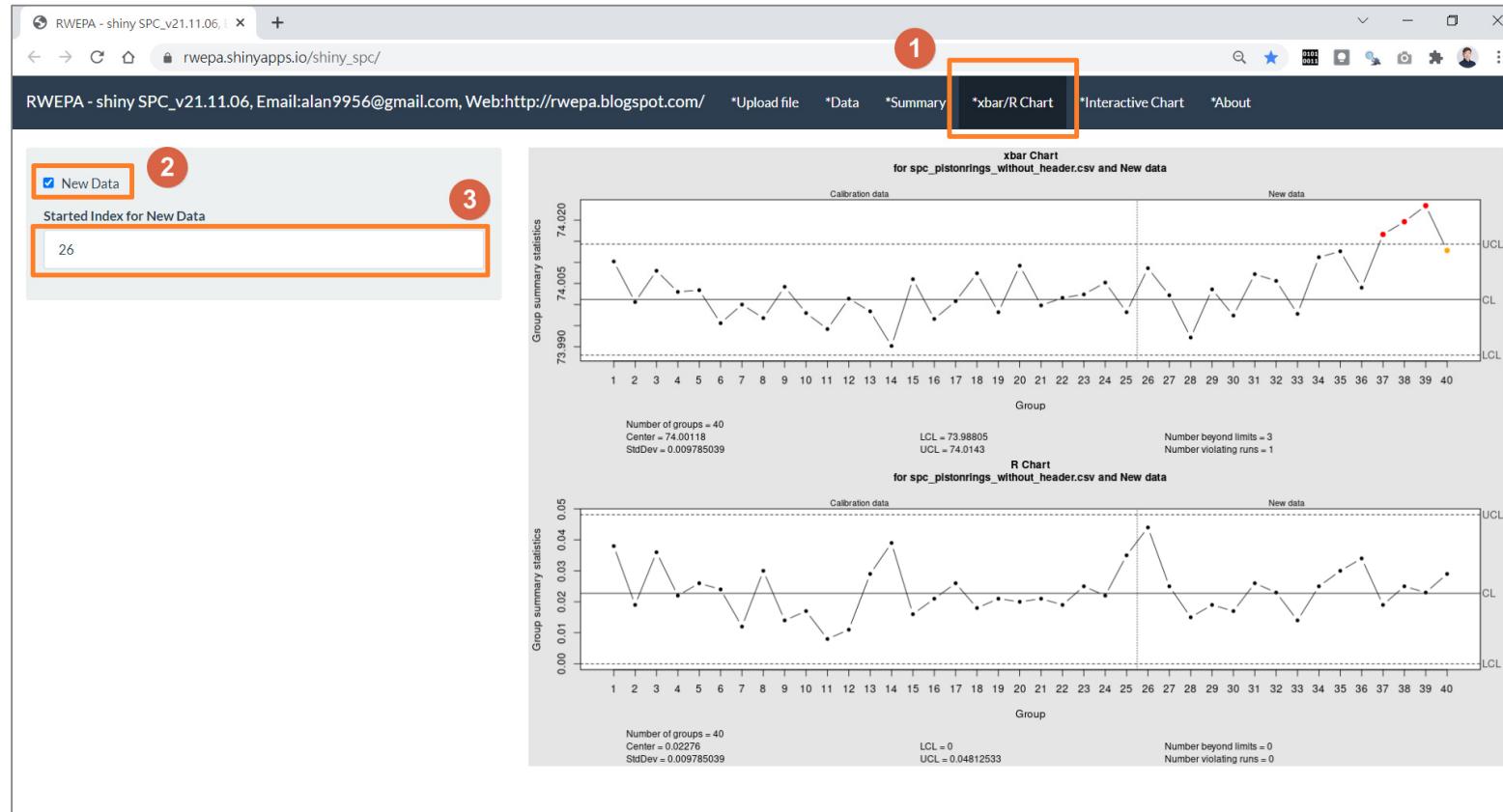
shiny 顧客連接分析

- <https://rwepa.shinyapps.io/shinyCustomerConnect/>



品質管制圖(quality control chart)應用

- <http://rwepa.blogspot.com/2021/10/r-shiny-quality-control-chart.html>



品質管制圖應用 (續)

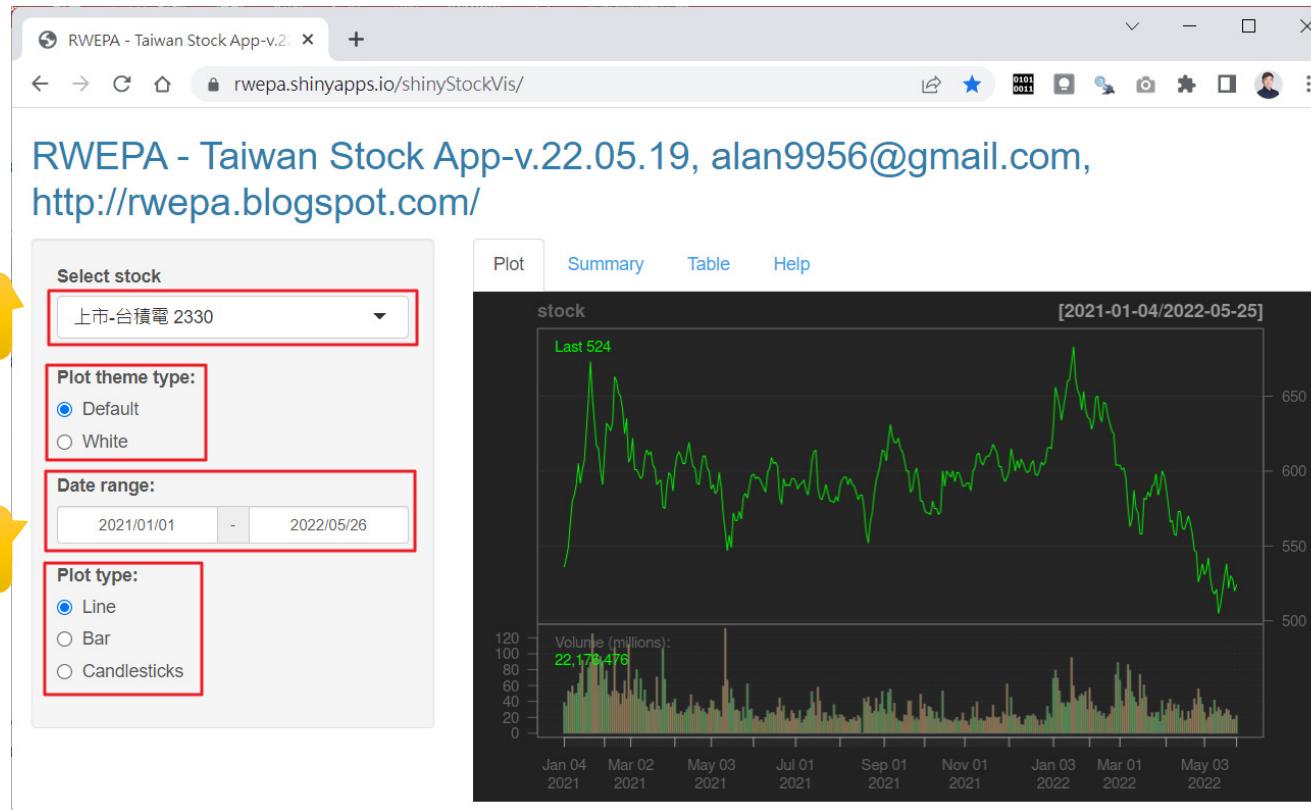
- https://github.com/rwepa/DataDemo/blob/master/spc_wafer_with_header.csv
- https://github.com/rwepa/DataDemo/blob/master/spc_pistonrings_without_header.csv

The screenshot shows a GitHub file page for 'spc_wafer_with_header.csv'. The page includes a header with 'master', the file path 'DataDemo / spc_wafer_with_header.csv', and buttons for 'Go to file' and '...'. Below this is a contributor section for 'rwepa' showing 'Add files via upload' and a 'Latest commit' from 14 days ago. It also shows '1 contributor'. The main content area displays the file's metadata: '26 lines (26 sloc) | 906 Bytes' and a search bar. To the right is a toolbar with buttons for 'Raw' (which is highlighted with a red box and arrow), 'Blame', and other file operations. Below the toolbar is a table preview of the CSV data:

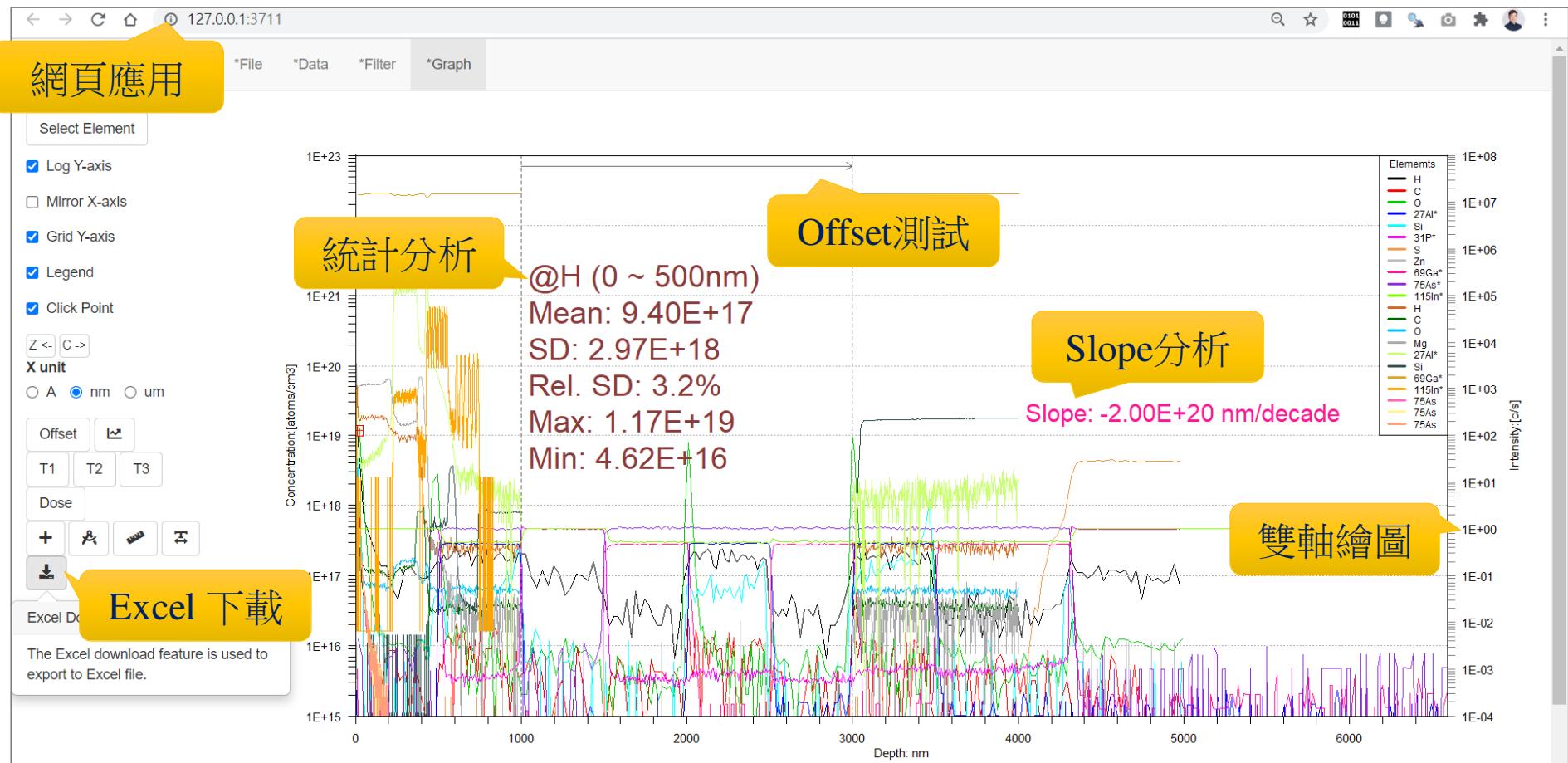
	x1	x2	x3	x4	x5
2	1.3235	1.4128	1.6744	1.4573	1.6914
3	1.4314	1.3592	1.6075	1.4666	1.6109
4	1.4284	1.4871	1.4932	1.4324	1.5674
5	1.5028	1.6352	1.3841	1.2831	1.5507
6	1.5604	1.2735	1.5265	1.4363	1.6441

Taiwan Stock App

- <https://rwepa.shinyapps.io/shinyStockVis/>
- <https://github.com/rwepa/DataDemo/blob/master/shinyStockVis.zip>



離子資料分析與視覺化應用





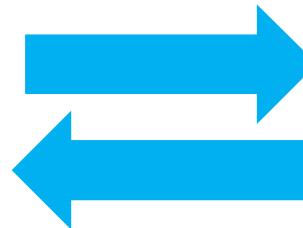
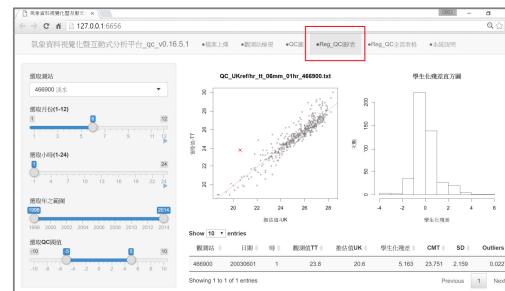
4.shiny 簡介



shiny 簡介

- 隨著資訊科技技術的進步，如何提供WEB化的應用服務。
- shiny 套件提供更方便，互動式與動態更新等應用
 - 輸入 - 文字方塊、下拉式選單、按鈕。
 - 處理 - 執行 R 運算、建立模型。
 - 輸出 - 網頁中呈現文字摘要、表格與圖表等。
- shiny 套件來自於R語言，目前已經可應用於Python語言。
- shiny for R 0.2.3 - December 01, 2012.
- shiny for Python 0.2.1 - RStudio's CTO [Joe Cheng](#) announced Shiny for Python, RStudio Conference, July 28, 2022.

shiny (server) 架構



ui.R



server.R



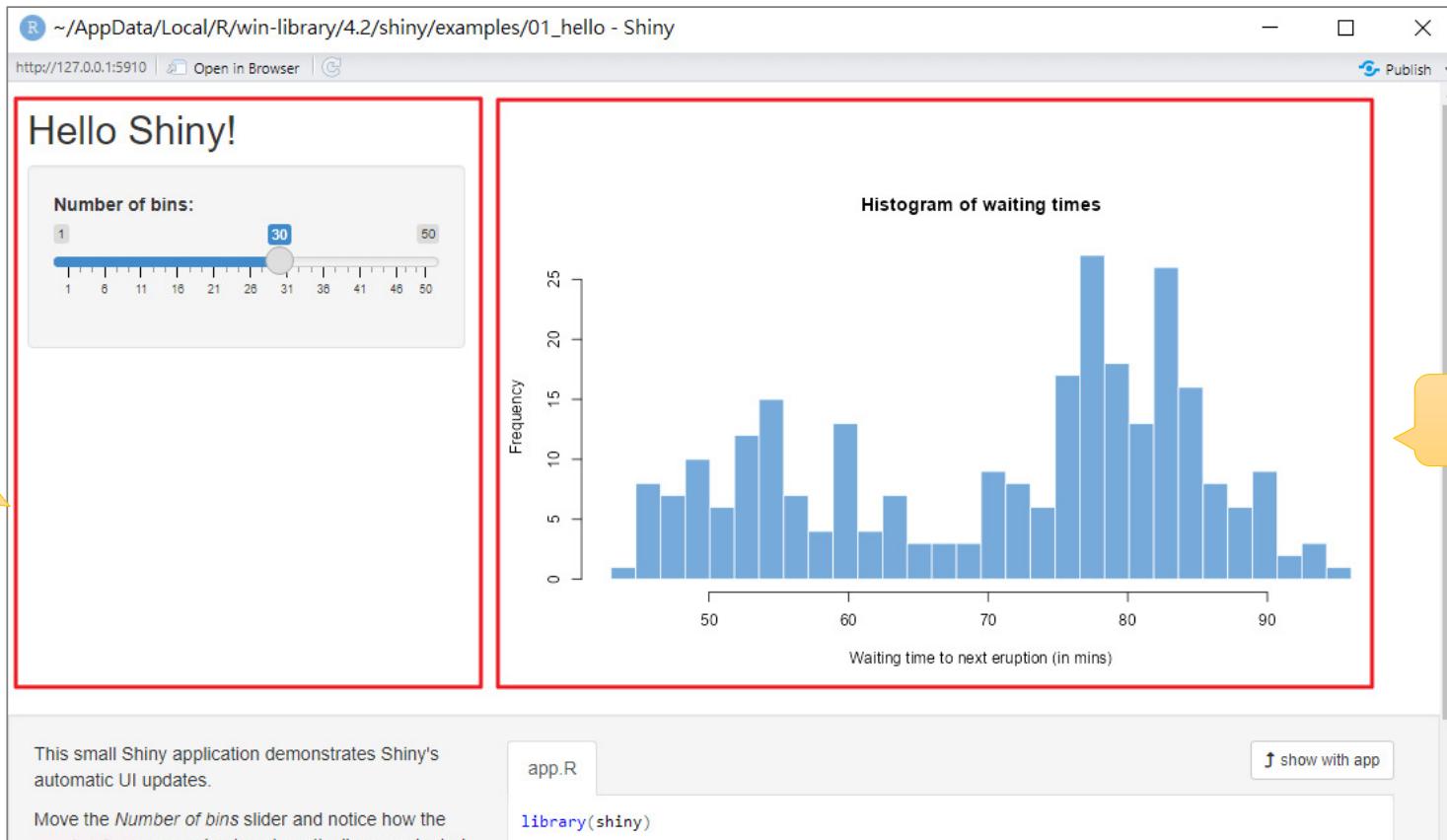
R – shiny tutorial

- Shiny 參考 第10章 - 10.基礎互動式shiny,進階互動式shiny Server佈署
 - PDF https://github.com/rwepa/DataDemo/blob/master/ai_using_python_and_r/ai_using_python_and_r_part2_r.pdf
 - R 下載 https://github.com/rwepa/DataDemo/blob/master/ai_using_python_and_r/ai_using_python_and_r_part2_r.R
- R shiny 內建11個範例:

```
> dir(paste0(.libPaths(), "/shiny/examples"))
[1] "01_hello"        "02_text"         "03_reactivity"  "04_mpg"          "05_sliders"
[6] "06_tabssets"    "07_widgets"     "08_html1"       "09_upload"      "10_download"
[11] "11_timer"
```

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

Listening on http://127.0.0.1:5910





R - CRAN shiny 套件

互動式網頁應用

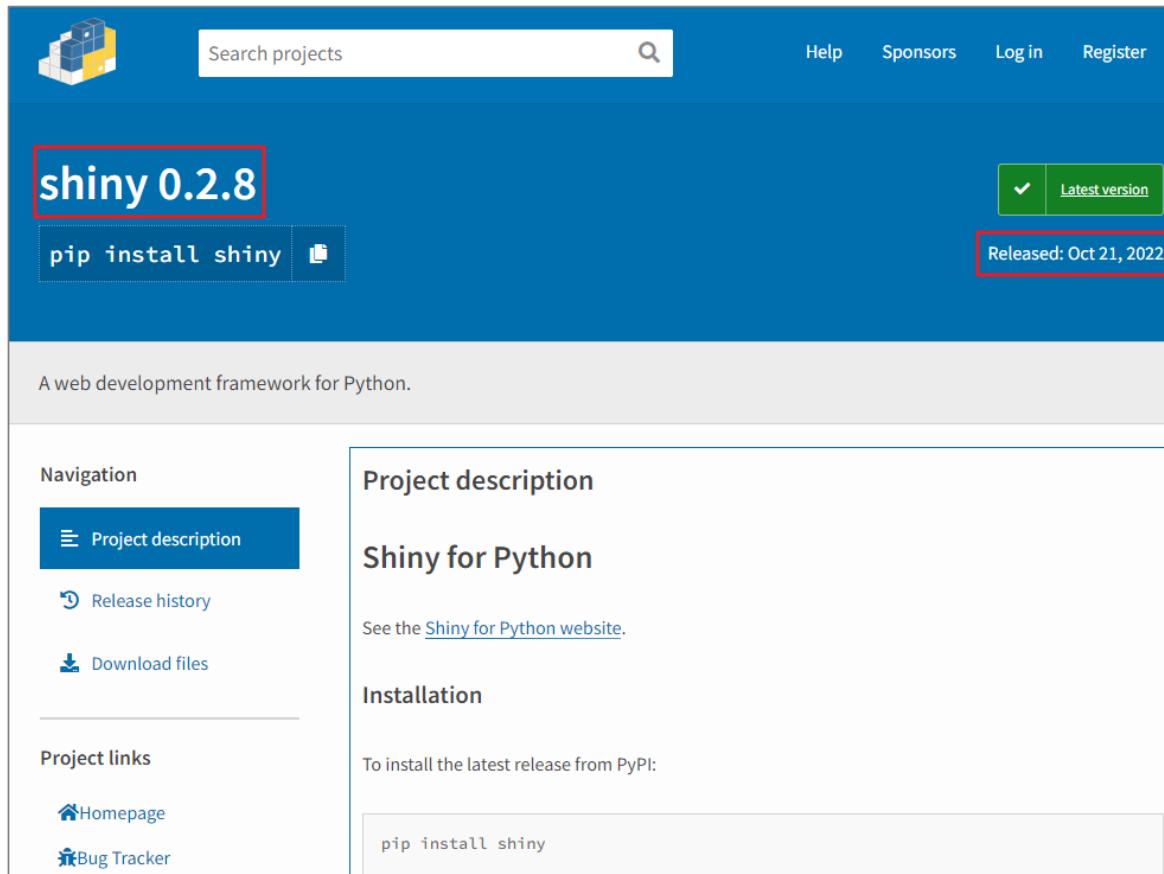
shiny: Web Application Framework for R

Makes it incredibly easy to build interactive web applications with R. Automatic "reactive" binding between inputs and outputs and extensive prebuilt widgets make it possible to build beautiful, responsive, and powerful applications with minimal effort.

Version:	1.7.2
Depends:	R (≥ 3.0.2), methods
Imports:	utils, grDevices, httpuv (≥ 1.5.2), mime (≥ 0.3), jsonlite (≥ 0.9.16), xtable , fontawesome (≥ 0.2.1), htmltools (≥ 0.5.2), R6 (≥ 2.0), sourcetools , later (≥ 1.0.0), promises (≥ 1.1.0), tools, crayon , rlang (≥ 0.4.10), fastmap (≥ 1.1.0), withr , commonmark (≥ 1.7), glue (≥ 1.3.2), bslib (≥ 0.3.0), cachem , ellipsis , lifecycle (≥ 0.2.0)
Suggests:	datasets, Cairo (≥ 1.5-5), testthat (≥ 3.0.0), knitr (≥ 1.6), markdown , rmarkdown , ggplot2 , reactlog (≥ 1.0.0), magrittr , yaml , future , dygraphs , ragg , showtext , sass
Published:	2022-07-19
Author:	Winston Chang  [aut, cre], Joe Cheng [aut], JJ Allaire [aut], Carson Sievert  [aut], Barret Schloerke  [aut], Yihui Xie [aut], Jeff Allen [aut], Jonathan McPherson [aut], Alan Dipert [aut], Barbara Borges [aut], RStudio [cph], jQuery Foundation [cph] (jQuery library and jQuery UI library), jQuery contributors [ctb, cph] (jQuery library; authors listed

<https://cran.r-project.org/web/packages/shiny/index.html>

Python – PyPI shiny 模組



The screenshot shows the PyPI project page for 'shiny'. At the top, the version '0.2.8' is highlighted with a red box. To its right, a green button indicates it's the 'Latest version'. Below this, a red box highlights the release date 'Released: Oct 21, 2022'. The main content area includes a brief description: 'A web development framework for Python.' On the left, a navigation sidebar lists 'Project description' (which is selected and highlighted in blue), 'Release history', and 'Download files'. Below that is a 'Project links' section with 'Homepage' and 'Bug Tracker' options.

A web development framework for Python.

Navigation

- [Project description](#)
- [Release history](#)
- [Download files](#)

Project links

- [Homepage](#)
- [Bug Tracker](#)

Project description

Shiny for Python

See the [Shiny for Python website](#).

Installation

To install the latest release from PyPI:

```
pip install shiny
```

<https://pypi.org/project/shiny/>

Shiny for Python

- <https://shiny.rstudio.com/py/>

The screenshot shows the Shiny for Python homepage. At the top is a blue navigation bar with the "Shiny for Python" logo and links for Docs, Install, Deploy, Shinylive, Gallery, Examples, and API. Below the bar is a large title "Shiny for Python". A paragraph explains that Shiny makes it easy to build interactive web applications with Python's data and scientific stack. A bulleted list highlights the application's approachability, flexibility, and performance. A red callout box contains a note that the application is in Alpha and may be unstable. At the bottom are two buttons: "Get Started" (highlighted with a red border) and "Examples". Below the buttons are three preview cards for different applications: "Elevation of mountain peaks", "How Does Regularization Strength Affect Coefficient Estimates?", and "CPU Usage %".

shiny

Docs

Install

Deploy

Shinylive

Gallery

Examples

API



Shiny for Python - Docs

The screenshot shows a web browser window with the title "Shiny for Python - Get started". The address bar contains the URL "shiny.rstudio.com/py/docs/get-started.html". The navigation bar includes tabs for "Shiny for Python", "Docs" (which is highlighted with a red box and has a red arrow pointing to it from the top), "Install", "Deploy", "Shinylive", "Gallery", "Examples", and "API". Below the navigation bar is a search bar with a magnifying glass icon and a user profile icon.

Get started

Learn Shiny in the browser (no installation needed!) or install on your computer.

Option 1: Learn in the browser

Example Shiny apps can run in your web browser.

You can skip installing Shiny, and go straight to the [next page](#).

Option 2: Install on your computer

```
pip install shiny  
shiny create my_app  
shiny run --reload my_app/app.py
```



Shiny for Python - Install

The screenshot shows a web browser window displaying the "Installing Shiny for Python" guide from shiny.rstudio.com. The browser's address bar shows the URL: shiny.rstudio.com/py/docs/install.html. The navigation bar includes links for "Shiny for Python", "Docs", "Install" (which is highlighted with a red box and has a red arrow pointing to it), "Deploy", "Shinylive", "Gallery", "Examples", and "API". The main content area features a large title "Installing Shiny for Python". Below the title, there is a paragraph of text: "To install Shiny for Python, first create a new directory for your first Shiny app, and change to it." Following this, there is a code block:

```
mkdir myapp  
cd myapp
```

A note below the code says: "Next, use either `pip` or `conda` to install the `shiny` package." There are two buttons: "Install with pip" (selected) and "Install with conda". A sidebar titled "On this page" contains a list of sections, with "Running" being the first item, followed by "Configure Visual Studio Code", both enclosed in a yellow box.

To install Shiny for Python, first create a new directory for your first Shiny app, and change to it.

```
mkdir myapp  
cd myapp
```

Next, use either `pip` or `conda` to install the `shiny` package.

Install with pip [Install with conda](#)

If you want to use a virtual environment, feel free to create/activate one now:

```
# Create a virtual environment in the .venv subdirectory  
python3 -m venv venv  
  
# Activate the virtual environment  
source venv/bin/activate
```



Shiny for Python - Deploy

Shiny for Python - Deploying Shiny [Docs](#) [Install](#) **Deploy** [Shinylive](#) [Gallery](#) [Examples](#) [API](#)

Deploying Shiny

(This page is about doing traditional Shiny application deployments. If you want to do a Shinylive deployment, see the [Shinylive](#) page.)

When it's time to put your Shiny app on the web, you can choose to deploy on your own servers or on our hosting service.

Deploy to shinyapps.io (cloud hosting)

The quickest way to get started is [shinyapps.io](#), which is our hosted service for deploying Shiny applications. With shinyapps.io, you don't need to set up a server; you just need to make an account on the site and then deploy the application there. Both free and paid tiers of service are available.

To use shinyapps.io, see the [shinyapps.io documentation](#).

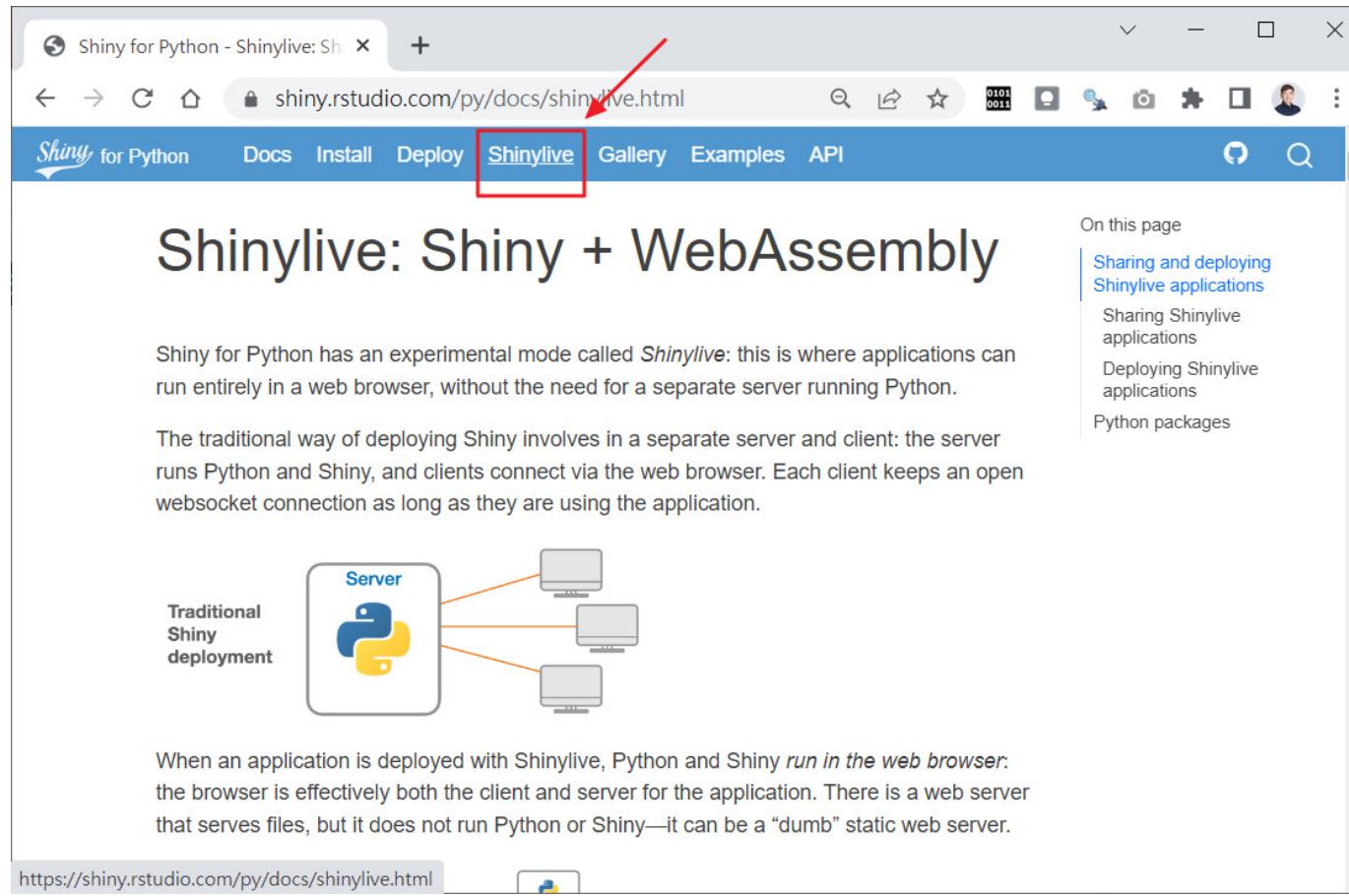
Deploy to Shiny Server (open source)

On this page

- [Deploy to shinyapps.io \(cloud hosting\)](#)
- Deploy to Shiny Server (open source)
- Deploy to RStudio Connect (commercial)
- Other hosting options



Shiny for Python - Shinylive



Shiny for Python - Shinylive: Shinylive | shiny.rstudio.com/py/docs/shinylive.html

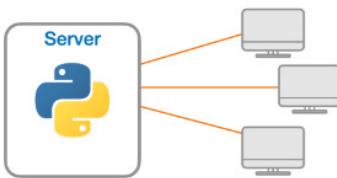
Shiny for Python Docs Install Deploy **Shinylive** Gallery Examples API

Shinylive: Shiny + WebAssembly

Shiny for Python has an experimental mode called *Shinylive*: this is where applications can run entirely in a web browser, without the need for a separate server running Python.

The traditional way of deploying Shiny involves in a separate server and client: the server runs Python and Shiny, and clients connect via the web browser. Each client keeps an open websocket connection as long as they are using the application.

Traditional Shiny deployment



When an application is deployed with Shinylive, Python and Shiny *run in the web browser*: the browser is effectively both the client and server for the application. There is a web server that serves files, but it does not run Python or Shiny—it can be a “dumb” static web server.

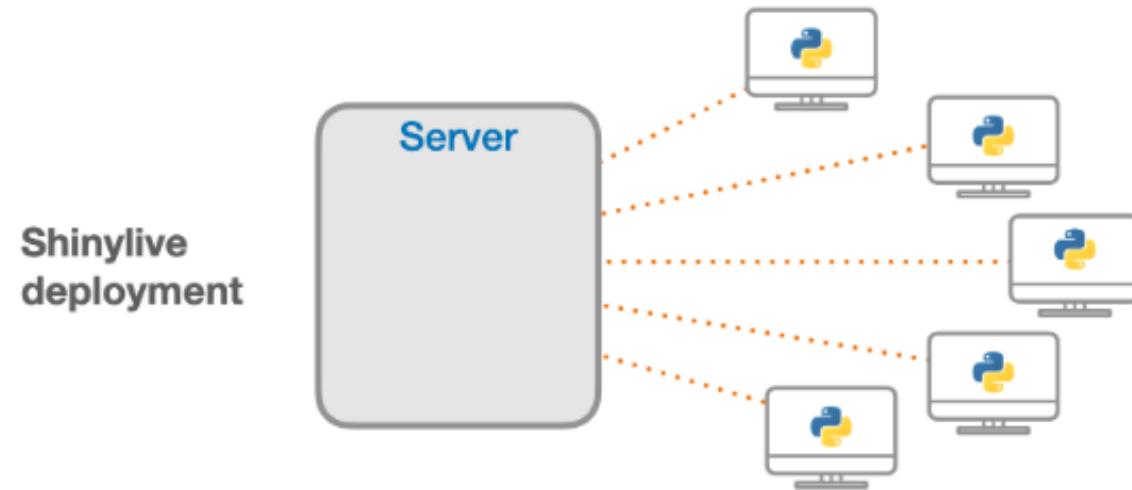
<https://shiny.rstudio.com/py/docs/shinylive.html>

On this page

- Sharing and deploying Shinylive applications
- Sharing Shinylive applications
- Deploying Shinylive applications
- Python packages

佈署 - shinylive

- shinylive 可以在用戶的 Web 瀏覽器中運行 Python，而不用在伺服端上運行 Python。Shinylive 的二大元件：
 - WebAssembly (wasm)：是一個編譯的二進位格式，可以在瀏覽器中以接近本機的速度執行，一般連結伺服端速度較慢。
 - Pyodide：提供許多相關模組，並將Python編譯為網頁組合 (WebAssembly) 。





Shiny for Python - Gallery

The screenshot shows the Shiny for Python Gallery page at shiny.rstudio.com/py/gallery/. A red arrow points to the "Gallery" tab in the navigation bar, which is highlighted in blue. The main content area displays several app examples:

- Airmass calculator**: Shows two plots of airmass over time for different objects (M1, M2, M3, PLX289) at specific coordinates (Latitude: 27.238446, Longitude: 27.548754). It includes a map of the location.
- Orbit simulation**: A 3D plot showing the orbital path of a target object (M1, NGC6, PLX289) around Earth (27.238446, 27.548754) with Moon (27.572348, 27.57218) and Sun (27.572348, 27.57218) reference points.
- Regularization strength**: A section titled "How Does Regularization Strength Affect Coefficient Estimates?" featuring a scatter plot of coefficient estimates for Lasso, Ridge, and Least Squares methods across different regularization strengths (α). It includes an explanation of regularization and a comparison of Lasso, Ridge, and Least Squares.
- CPU usage monitor**: A simple visualization showing CPU Usage % over time.
- Shiny Wordle**: A wordle game.
- Simulate data for a t-test**: A tool for generating simulated data for statistical testing.

Each app example has "View app" and "View source" links below it.



Shiny for Python - Examples

The screenshot shows the Shiny for Python Examples interface in a browser. The sidebar on the left lists examples under 'Basic' and 'Featured' categories. The main area contains a code editor with two files: 'app.py' and 'server.py'. The 'app.py' file contains:

```
from shiny import App, render, ui

app_ui = ui.page_fluid(
    ui.input_slider("n", "RWEPA測試", 0, 100, 20),
    ui.output_text_verbatim("txt"),
)

def server(input, output, session):
    @output
    @render.text
    def txt():
        return f"n*2 is {input.n() * 2}"
```

The 'server.py' file contains:

```
>>> 1+2
3
>>> from shiny import App, render, ui
>>> app_ui = ui.page_fluid(
    ui.input_slider("n", "N", 0, 100, 20),
    ui.output_text_verbatim("txt"),
)
>>>
```

The preview window on the right shows a slider set to 20, with the text output below it reading "n*2 is 40".

1. 範例選單
2. 程式碼編輯區
3. 控制台 (Console)
4. 執行App
(Ctrl + Shift + Enter)
5. 執行 shiny 結果



Shiny for Python - API

The screenshot shows a web browser displaying the Shiny for Python API reference. The URL is shiny.rstudio.com/py/api/#page-containers. The page is titled "Page containers".

- Left Sidebar (1):** A list of UI components:
 - shiny.ui.page_navbar
 - shiny.ui.page_fluid
 - shiny.ui.page_fixed
 - shiny.ui.page_bootstrap
 - shiny.ui.layout_sidebar
 - shiny.ui.panel_sidebar
 - shiny.ui.panel_main
 - shiny.ui.column
 - shiny.ui.row
 - shiny.ui.input_select
 - shiny.ui.input_selectize
 - shiny.ui.input_slider
- Main Content Area (2):** A table of contents for "Page containers".

<code>ui.page_navbar(*args[, title, id, selected, ...])</code>	Create a navbar with a navs bar and a title.
<code>ui.page_fluid(*args[, title, lang])</code>	Create a fluid page.
<code>ui.page_fixed(*args[, title, lang])</code>	Create a fixed page.
<code>ui.page_bootstrap(*args[, title, lang])</code>	Create a Bootstrap UI page container.
- Right Sidebar (3):** A list of API categories:
 - API Reference Intro
 - API Reference** (highlighted)
 - Page containers
 - UI Layout
 - UI Inputs
 - Update inputs
 - Navigation (tab) panels
 - UI panels
 - Uploads & downloads
 - Custom UI
 - Rendering outputs
 - Reactive programming
 - Create and run applications
 - Display messages
 - Error validation
 - Modules
 - Type hints
 - Developer facing tools

1. 左側列出所有函數
2. 中間為函數說明
3. 右側API -17項:
Page containers ...
Developer facing tools.



線上說明 - shiny.ui.page_navbar

The screenshot shows a browser window displaying the API documentation for the `shiny.ui.page_navbar` function. The page has a header with the Shiny logo and navigation links. On the left, there's a sidebar with a search bar and a list of related functions. The main content area is titled `shiny.ui.page_navbar` and contains the following text:

```
shiny.ui.page_navbar(*args, title=None, id=None, selected=None, position='static-top', header=None, footer=None, bg=None, inverse=False, collapsible=True, fluid=True, window_title=<shiny.types.MISSING_TYPE object>, lang=None)
```

Create a navbar with a navs bar and a title.

Parameters:

- `args` (`NavSetArg`) – UI elements.
- `title` (`Union[str, Tag, TagList, None]`) – The browser window title (defaults to the host URL of the page). Can also be set as a side effect via `panel_title()`.
- `id` (`Optional[str]`) – If provided, will create an input value that holds the currently selected nav item.
- `selected` (`Optional[str]`) – Choose a particular nav item to select by default value (should match its `value`).

畫面好像是 scikit-learn
線上說明



5.shiny for Python 實作篇1 - my_app

3行指令建立 my_shiny

- pip install shiny (或 conda install -c conda-forge shiny)
- shiny create my_app
- shiny run --reload my_app/app.py

localhost:8000

Hello Shiny!

N

0 20 100

n*2 is 40

http://localhost:8000/





app.py 包括四大區塊

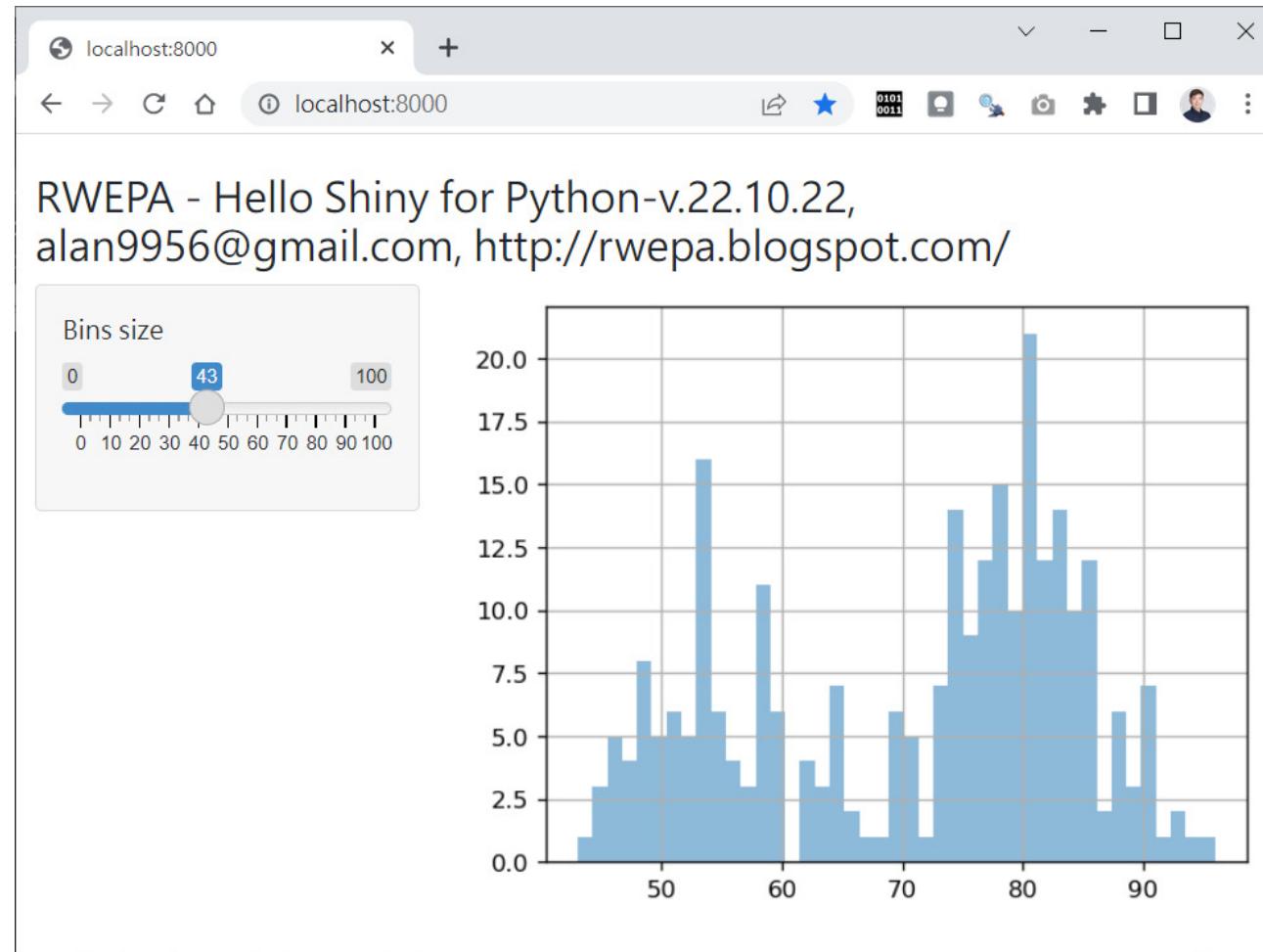
```
app.py x
1 from shiny import App, render, ui
2
3 app_ui = ui.page_fluid(
4     ui.h2("Hello Shiny!"),
5     ui.input_slider("n", "N", 0, 100, 20),
6     ui.output_text_verbatim("txt"),
7 )
8
9
10 def server(input, output, session):
11     @output
12     @render.text
13     def txt():
14         return f"n*2 is {input.n() * 2}"
15
16
17 app = App(app_ui, server)
```

1. 載入模組
2. 宣告 ui
3. 宣告 server
4. 執行 App



6.shiny for Python 實作篇2 – 01_hello

Hello Shiny for Python





pyshiny_01_hello.py

```
13 from shiny import App, render, ui
14 from rdatasets import data
15 # datasets list: https://vincentarelbundock.github.io/Rdatasets/datasets.html
16
17 df = data("faithful")
18
19 app_ui = ui.page_fluid(
20     ui.h3("RWEPA - Hello Shiny for Python-v.22.10.22, alan9956@gmail.com, http://rwepa.blogspot.com/"),
21     ui.layout_sidebar(
22         ui.panel_sidebar(ui.input_slider("bins", "Bins size", min=0, max=100, value=30)),
23         ui.panel_main(ui.output_plot("plot")),
24     ),
25 )
26
27 def server(input, output, session):
28     @output
29     @render.plot(alt="A histogram")
30     def plot():
31
32         # df['waiting'].plot(kind='hist', alpha=0.5, bins=30, title='Histogram of waiting times', grid=True)
33         myplot = df['waiting'].plot(kind='hist', alpha=0.5, bins=input.bins(), title='Histogram of waiting times', grid=True)
34         return myplot
35
36 app = App(app_ui, server)
```

The diagram illustrates the data flow in the code. A blue box highlights the 'bins' parameter in the `ui.input_slider` call at line 22. A blue arrow points from this box to the corresponding parameter in the `plot` function definition at line 33, specifically to the line `bins=input.bins()`.

Hello Shiny for Python – 執行

- d:
- cd shinydemo
- shiny run --reload pyshiny_01_hello/pyshiny_01_hello.py
- 瀏覽器: <http://localhost:8000/>

```
命令提示字元 - shiny run --reload pyshiny_01_hello/pyshiny_01_hello.py
Microsoft Windows [版本 10.0.19044.2130]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。

C:\Users\asus>d:
D:\>cd shinydemo
D:\shinydemo>shiny run --reload pyshiny_01_hello/pyshiny_01_hello.py
[32mINFO [0m: Will watch for changes in these directories: ['D:\\shinydemo\\pyshiny_01_hello']
[32mINFO [0m: Uvicorn running on [1mhttp://127.0.0.1:8000 50m /PyCharm CTRL+C to quit]
[32mINFO [0m: Started reloader process [ 36m [1m
[32mINFO [0m: Started server process [ 36m13696
[32mINFO [0m: Waiting for application startup.
[32mINFO [0m: Application startup complete.
```

檔案架構

本機 > Data (D:) > shinydemo > pyshiny_01_hello

__pycache__

pyshiny_01_hello.py



7.結論

結論

- shiny 功能: 產銷人發財, 學術, Python研究
- shiny + Python
- 實作範例1 - my_app
- 實作範例2 - 01_hello

參考資料

1. RWEPA: <http://rwepa.blogspot.com/>
2. RWEPA shiny: <http://rwepa.blogspot.com/search?q=shiny>
3. Shiny for R: <https://shiny.rstudio.com/>
4. Shiny for Python: <https://shiny.rstudio.com/py/>
5. shiny for Python 實作篇2 – 01_hello 程式碼:
https://github.com/rwepa/shiny_python





謝謝您的聆聽

Q & A



李明昌

alan9956@gmail.com

<http://rwepa.blogspot.tw/>