

國立高雄科技大學智慧商務系

109 年學年度研究報告

水庫資料爬蟲及數據視覺化

第壹章、緒論.....	2
第壹章、第一節 前言.....	2
第壹章、第二節 研究動機、目的與目標.....	2
第壹章、第三節 研究流程.....	2
第壹章、第四節 開發軟硬體設備.....	3
第壹章、第五節 計畫時程.....	3
第貳章、系統設計與架構.....	4
第貳章、第一節 系統與程式架構簡圖.....	4
第參章、系統實作與展示.....	6
第參章、第一節 原始資料、資料爬蟲與更新置資料庫.....	6
第參章、第二節 資料庫存儲資料.....	9
第參章、第三節 常駐定時爬蟲.....	11
第參章、第四節 從 SQL 中查詢資料.....	12
第參章、第五節 取得資料與繪圖.....	13
第參章、第六節 GUI.....	15
第參章、第七節 結果.....	16
第肆章、結論與未來展望.....	24
第伍章、參考文獻.....	24
附錄 程式碼.....	25

摘要

臺灣是一個位處熱帶與亞熱帶的國家，擁有全球數一數二多的降雨量，照理講，我們應該不會缺水。但處在臺灣的我們，常常面臨缺水問題，甚至是世界排名第十八位的缺水國家。由於臺灣地區地狹人稠、山坡陡峭、雨勢集中再加上河川短促，所以大部分的水資源都流向海洋。因此，為了更有效的利用水資源，水庫對我們來說就是一個很重要的建設。

而今年 4 月、5 月的乾旱，造成民眾的不便。這使台灣陷入 56 年來最嚴重的乾旱，許多水庫的蓄水量甚至低於 30%，甚至，我的家鄉——台中還因此面臨「供五停二」的情形。這使得民眾突然都開始關心起了水情，像是觀看線上水庫直播。

我們知道水庫的資料都是公開在網路上的，於是便著手設計爬蟲程式，爬下了 6/7 至今的蓄水量資料，最後再使用 matplotlib 將統計出的資料視覺化。

關鍵詞：水情資料、爬蟲、Request、Beautifulsoup4、Matplotlib、PyMySQL、MySQL

第壹章、緒論

壹、第一節 前言

水庫具有許多功能如:發電、灌溉、供水、防洪、航運、養殖、旅遊、攔蓄泥沙等，但這其中與水位卻息息相關，像是，水位太低時無法發電及灌溉，水位太高會無法起到防洪的作用。

壹、第二節 研究動機、目的與目標

經網路搜尋後，多看到的是當日的水庫資料，公開的歷史資料大多都是很簡陋的資料表，無法一眼了解水位的趨勢。本次專題將針對公開在水利署網站上的公開資料進行資料爬取，希望使用者能在使用時，一眼就能透過數據視覺化後的圖表來了解近日的趨勢。

壹、第三節 研究流程



圖一：研究流程圖

(圖一資料來源：研究者自行繪製)

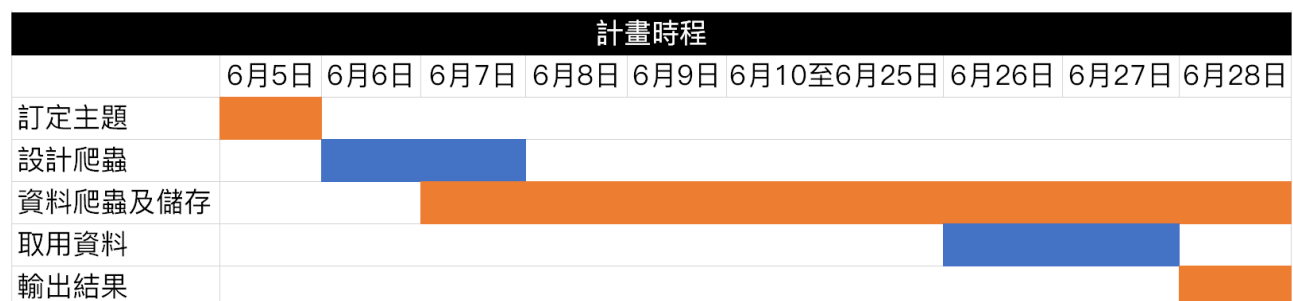
1. 訂定主題：選擇題目。
2. 設計爬蟲：設計出可爬取資料的程式。
3. 資料爬蟲：爬取資料。

4. 資料儲存：將結果存回資料庫。
5. 取用資料：將資料從資料庫中取出。
6. 輸出結果：將資料繪製成圖表。

壹、第四節 開發軟硬體設備

硬體	GCP(f1-micro)、PC
作業系統	Ubuntu 20.04 server 、Windows 10
瀏覽器	Brave
資料庫軟體	MySql 8.0.25、phpMyAdmin 4.9.5、PyMySQL
開發軟體	VisulStudioCode
開發語言及套件	Python 3.9.5：(Request、BeautifulSoup4、Matplotlib、Tkinter)、SQL

壹、第五節 計畫時程

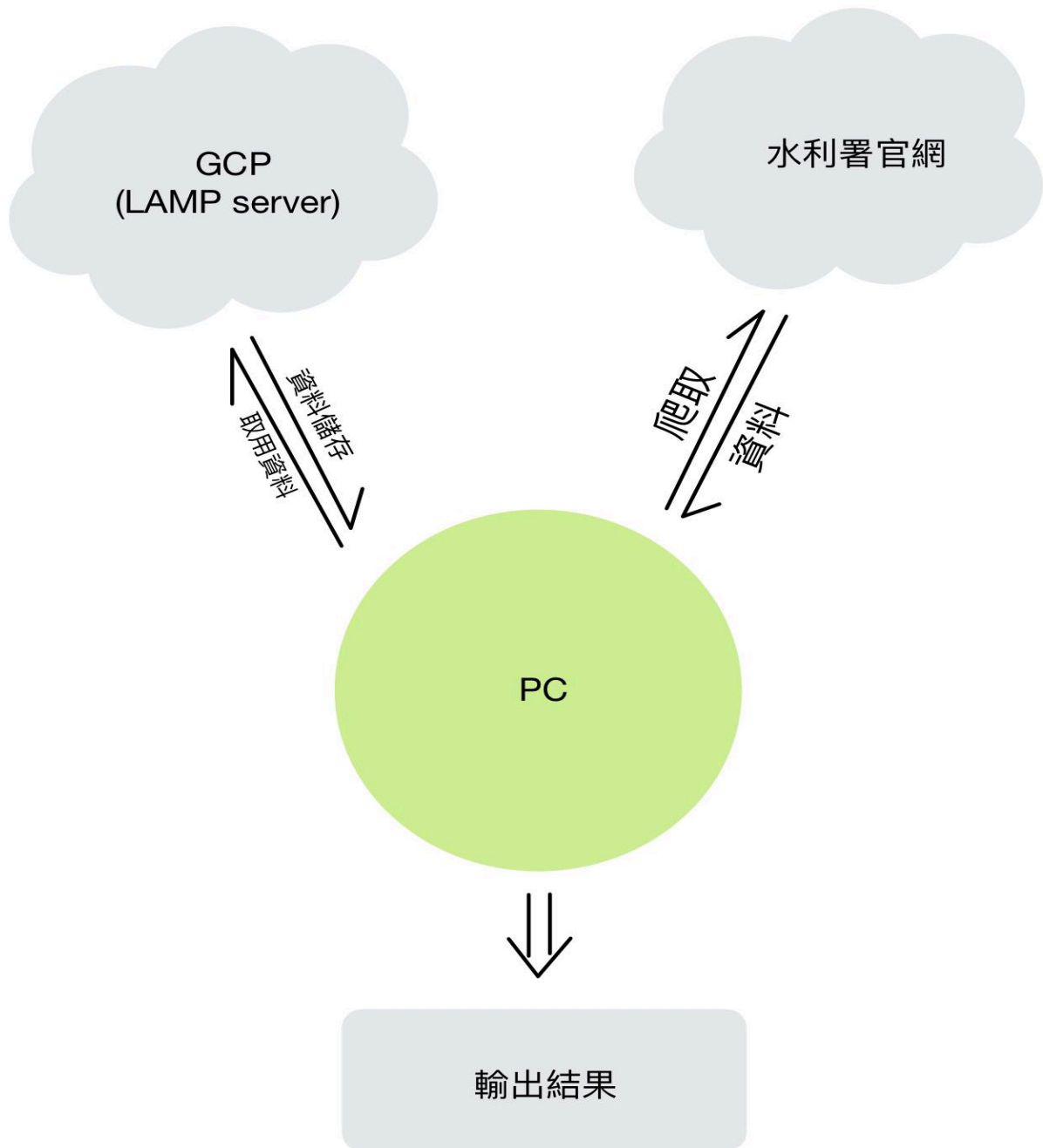


圖二：計畫時程甘特圖

(圖二資料來源：研究者自行繪製)

第貳章、系統設計與架構

貳、第一節 系統與程式架構簡圖



圖三：系統架構圖

(圖三資料來源：研究者自行繪製)

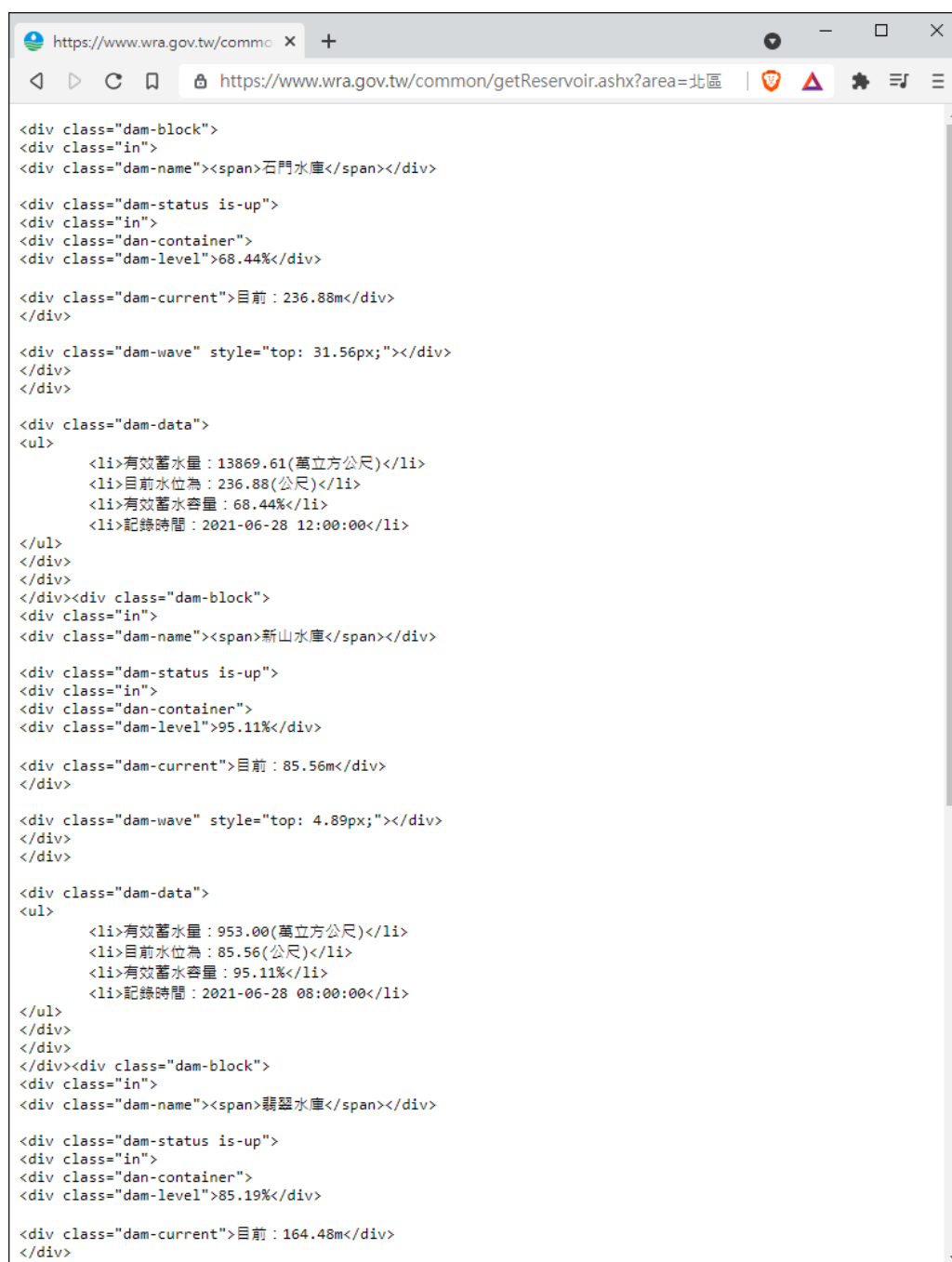


圖四：程式架構圖

(圖四資料來源：研究者自行繪製)

第參章、系統實作與展示

第參章、第一節 原始資料、資料爬蟲與更新置資料庫



```
<div class="dam-block">
<div class="in">
<div class="dam-name"><span>石門水庫</span></div>

<div class="dam-status is-up">
<div class="in">
<div class="dam-container">
<div class="dam-level">68.44</div>

<div class="dam-current">目前：236.88m</div>
</div>

<div class="dam-wave" style="top: 31.56px;"></div>
</div>

<div class="dam-data">
<ul>
<li>有效蓄水量：13869.61(萬立方公尺)</li>
<li>目前水位為：236.88(公尺)</li>
<li>有效蓄水容量：68.44</li>
<li>記錄時間：2021-06-28 12:00:00</li>
</ul>
</div>
</div><div class="dam-block">
<div class="in">
<div class="dam-name"><span>新山水庫</span></div>

<div class="dam-status is-up">
<div class="in">
<div class="dam-container">
<div class="dam-level">95.11</div>

<div class="dam-current">目前：85.56m</div>
</div>

<div class="dam-wave" style="top: 4.89px;"></div>
</div>

<div class="dam-data">
<ul>
<li>有效蓄水量：953.00(萬立方公尺)</li>
<li>目前水位為：85.56(公尺)</li>
<li>有效蓄水容量：95.11</li>
<li>記錄時間：2021-06-28 08:00:00</li>
</ul>
</div>
</div><div class="dam-block">
<div class="in">
<div class="dam-name"><span>翡翠水庫</span></div>

<div class="dam-status is-up">
<div class="in">
<div class="dam-container">
<div class="dam-level">85.19</div>

<div class="dam-current">目前：164.48m</div>
</div>
</div>
```

圖五：原始資料

原始資料將全臺灣水庫分為北區、中區、南區，故爬取 3 次分別對應，將資料取回


```

5  def reservoir(area):
6      # 爬取資料
7      data = {"area": f"{area}"}
8      response = requests.get("https://www.wra.gov.tw/common/getReservoir.ashx", data)
9      soup = BS(response.text, "html.parser")
10     # 區域偵測 Area detection
11     currentarea = data["area"]
12     print("目前" + f"{currentarea}" + "水庫:")
13     # 列出目前水庫資料(有效蓄水量, 目前水位, 目前蓄水容量, 記錄時間)
14     damdata = []
15     for ul in soup.find_all("ul"):
16         for li in ul.find_all("li"):
17             damdata.append(li.string)
18     # 資料長度
19     releng = len(soup.find_all("ul"))
20     print("共有" + f"{releng}" + "個水庫")
21     dataleng = releng * 4
22     # 列出所有搜尋到的水庫
23     RSN = []
24     span_tags = soup.find_all("span")
25     for tag in span_tags:
26         RSN.append(tag.string)
27     # 擷取有效蓄水量, Effective storage capacity
28     ESC = []
29     for i in range(0, dataleng, 4):
30         ESC.append(str(damdata[i]).replace("有效蓄水量:", "").replace("(萬立方公尺)", ""))
31     # 擷取目前水位, Current water Level
32     CWL = []
33     for i in range(1, dataleng, 4):
34         CWL.append(str(damdata[i]).replace("目前水位為:", "").replace("(公尺)", ""))
35     # 擷取目前蓄水容量, Current storage capacity
36     CSC = []
37     for i in range(2, dataleng, 4):
38         CSC.append(str(damdata[i]).replace("有效蓄水容量:", ""))
39     # 擷取記錄時間, Record time
40     RT = []
41     for i in range(3, dataleng, 4):
42         RT.append(str(damdata[i]).replace("記錄時間:", ""))
43     # 回傳資料
44     result = [RSN, ESC, CWL, CSC, RT]
45     return result

```

圖六：爬蟲 code

分別爬取水庫名稱、有效蓄水量、目前水位高度，目前蓄水容量及記錄時間。

```

6 def Update():
7     date = datetime.now().strftime("%Y-%m-%d") # 時間, Time
8     RSN = [] # 水庫名稱, Reservoir name
9     ESC = [] # 有效蓄水量, Effective storage capacity
10    CWL = [] # 目前水位, Current water Level
11    CSC = [] # 目前蓄水容量, Current storage capacity
12    RT = [] # 記錄時間, Record time
13
14    # 取得所有水庫資料
15    data = ["北區", "中區", "南區"]
16    for i in data:
17        result = RS.reservoir(f"{i}")
18        RSN += result[0]
19        ESC += result[1]
20        CWL += result[2]
21        CSC += result[3]
22        RT += result[4]
23
24    # 寫入資料庫
25    conn = pymysql.connect(
26        host="xiaoyi1211.ddns.net",
27        port=3306,
28        user="chy",
29        passwd="Henrychy@1211",
30        db="reservoir",
31        charset="utf8",
32    )
33    cursor = conn.cursor(pymysql.cursors.DictCursor)
34
35    # 更新所有資料
36    print("正在更新總資料:")
37    for i in range(len(RSN)):
38        sql = f"UPDATE `dam-data` SET `ESC`='{str(ESC[i])}', `CWL`='{str(CWL[i])}', `CSC`='{str(CSC[i])}', `RT`='{str(RT[i])}' WHERE `RN`='{str(RSN[i])}'"
39        cursor.execute(sql)
40        print(f"第{i+1}筆資料")
41    conn.commit()
42
43    # 創建當日資料表
44    sql = f"CREATE TABLE IF NOT EXISTS `{date}` (`RN` varchar(20) NOT NULL, `CSC` varchar(20) NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;"
45    try:
46        cursor.execute(sql)
47    except Exception as e:
48        print("Mysql Error %d: %s" % (e.args[0], e.args[1]))
49        pass
50
51    # 更新水位資訊
52    sql = f"SELECT * FROM `{date}`"
53    cursor.execute(sql)
54    if cursor.fetchall() != ():
55        print("正在更新水位資料:")
56        for i in range(len(RSN)):
57            sql = f"UPDATE `{date}` SET `CSC`='{str(CSC[i])}' WHERE `RN` = '{str(RSN[i])}'"
58            cursor.execute(sql)
59            print(f"第{i+1}筆資料")
60        conn.commit()
61    else:
62        print("正在新增水位資料:")
63        for i in range(len(RSN)):
64            sql = f"INSERT INTO `{date}` (`RN`, `CSC`) VALUES ('{str(RSN[i])}', '{str(CSC[i])}')"
65            cursor.execute(sql)
66            print(f"第{i+1}筆資料")
67        conn.commit()
68    message = "已更新完成"
69    cursor.close()
70    conn.close()
71    return message

```

圖七、八：更新至資料庫

對應圖五分區問題，故執行爬蟲 3 次，分別將北區、中區、南區放入。由於 GCP 的 IP 是變動的，在 host 這邊使用動態 DNS 代替 IP 位址。而更新水位資訊則是判斷，當日的資料表是否存在，如不存在，就創建一張資料表。

第參章、第二節 資料庫存儲資料

mysql

performance_schema

phpmyadmin

reservoir

新增

2021-06-07

2021-06-08

2021-06-09

2021-06-10

2021-06-11

2021-06-12

2021-06-13

2021-06-14

2021-06-15

2021-06-16

2021-06-17

2021-06-18

2021-06-19

2021-06-20

2021-06-21

2021-06-22

2021-06-23

2021-06-24

2021-06-25

2021-06-26

2021-06-27

2021-06-28

dam-data

sys

wordpress

資料表

2021-06-07

2021-06-08

2021-06-09

2021-06-10

2021-06-11

2021-06-12

2021-06-13

2021-06-14

2021-06-15

2021-06-16

2021-06-17

2021-06-18

2021-06-19

2021-06-20

2021-06-21

2021-06-22

2021-06-23

2021-06-24

2021-06-25

2021-06-26

2021-06-27

2021-06-28

dam-data

23 張資料表

RN

CSC

石門水庫

68.44%

新山水庫

95.11%

翡翠水庫

85.19%

寶山第二水庫

93.59%

永和山水庫

44.11%

明德水庫

64.99%

鯉魚潭水庫

58.20%

德基水庫

31.67%

日月潭水庫

96.79%

湖山水庫

99.78%

仁義潭水庫

96.47%

烏山頭水庫

76.25%

曾文水庫

47.80%

南化水庫

100.00%

阿公店水庫

0.79%

牡丹水庫

76.38%

圖九、十：資料庫概覽 及 當日資料表

RN	ESC	CWL	CSC	RT
石門水庫	13869.61	236.88	68.44%	2021-06-28 12:00:00
新山水庫	953.00	85.56	95.11%	2021-06-28 08:00:00
翡翠水庫	28581.42	164.48	85.19%	2021-06-28 12:00:00
寶山第二水庫	2945.46	148.64	93.59%	2021-06-28 12:00:00
永和山水庫	1322.74	73.1100006103516	44.11%	2021-06-28 12:00:00
明德水庫	794.24	57.92	64.99%	2021-06-28 07:00:00
鯉魚潭水庫	6662.09	287.22	58.20%	2021-06-28 12:00:00
德基水庫	5903.95	1362.87	31.67%	2021-06-28 07:00:00
日月潭水庫	12395.06	747.97	96.79%	2021-06-28 07:00:00
湖山水庫	5075.33	211.45	99.78%	2021-06-28 12:00:00
仁義潭水庫	2438.30	104.61	96.47%	2021-06-28 07:00:00
烏山頭水庫	5992.00	56.16	76.25%	2021-06-28 07:00:00
曾文水庫	24359.00	214.05	47.80%	2021-06-28 12:00:00
南化水庫	9210.16	180.22	100.00%	2021-06-28 07:00:00
阿公店水庫	12.00	29.31	0.79%	2021-06-28 12:00:00
牡丹水庫	2025.00	137.25	76.38%	2021-06-28 12:00:00

圖十一：總資料表

最初設計時，爬取有效蓄水量是為了統計水庫可用容量，但其實資料變動不大，因此後來便取消分析有效蓄水量。而水位高度則是與蓄水容量功能相似。紀錄時間則是為了觀察水庫的更新頻率，部分水庫每 1 小時會更新一次資訊，而部分水庫則是每日早上 7 時才會更新一次。

第參章、第三節 常駐定時爬蟲

```
1 import threading
2 import reservoir_update as up
3
4
5 def set_interval(func, sec):
6     def func_wrapper():
7         set_interval(func, sec)
8         func()
9
10    t = threading.Timer(sec, func_wrapper)
11    t.start()
12    return t
13
14
15 set_interval(print(up.Update()), 3600)
16
```

C:\Users\henry\Documents\GitHub\PythonLearning\updateTimer.exe

第6筆資料
第7筆資料
第8筆資料
第9筆資料
第10筆資料
第11筆資料
第12筆資料
第13筆資料
第14筆資料
第15筆資料
第16筆資料
正在更新水位資料:
第1筆資料
第2筆資料
第3筆資料
第4筆資料
第5筆資料
第6筆資料
第7筆資料
第8筆資料
第9筆資料
第10筆資料
第11筆資料
第12筆資料
第13筆資料
第14筆資料
第15筆資料
第16筆資料
已更新完成

圖十二：定時器程式與執行畫面

這裡使用 pyinstaller 將定時器打包成 exe 檔，如此一來，便不必為了爬蟲還要開

VisulStudioCode 了。在此則引入更新至資料庫的程式，以每小時一次的頻率執行。

第參章、第四節 從 SQL 中查詢資料

```
4 def selsincsc(date, rn):
5     data = []
6 >     conn = pymysql.connect( ...
13 )
14     cursor = conn.cursor(pymysql.cursors.DictCursor)
15     cursor.execute(f'SELECT `CSC` FROM `{date}` WHERE RN = "{rn}"')
16     row = ""
17     row = cursor.fetchall()
18     rows = row[0]
19     value = rows["CSC"]
20     data = float(value.replace("%", ""))
21     conn.commit()
22     cursor.close()
23     conn.close()
24     return data
25
26
27 def showtable():
28     data = []
29 >     conn = pymysql.connect( ...
36 )
37     cursor = conn.cursor(pymysql.cursors.DictCursor)
38     cursor.execute('SHOW TABLES WHERE Tables_in_reservoir != "dam-data"')
39     row = ""
40     row = cursor.fetchall()
41     for i in row: # 取出value
42         for key, value in i.items():
43             data.append(value)
44     conn.commit()
45     cursor.close()
46     conn.close()
47     return data
48
49
50 def shownrn():
51     data = []
52 >     conn = pymysql.connect( ...
59 )
60     cursor = conn.cursor(pymysql.cursors.DictCursor)
61     cursor.execute("SELECT `RN` FROM `dam-data`")
62     row = ""
63     row = cursor.fetchall()
64     for i in row: # 取出value
65         for key, value in i.items():
66             data.append(value)
67     conn.commit()
68     cursor.close()
69     conn.close()
70     return data
```

圖十三：定時器程式與執行畫面

分別將不同的功能包裝成不同的函式，方便後續取用。

第參章、第五節 取得資料與繪圖

```
6   day = timedelta(hours=24)
7   date = datetime.now().strftime("%Y-%m-%d")
8
9
10  def getSinCSC(date1, date2, rn):
11      print(f"正在取得 {rn} {date1} 至 {date2} 的資料:")
12      date1 = datetime.strptime(date1, "%Y-%m-%d")
13      date2 = datetime.strptime(date2, "%Y-%m-%d")
14      date2 += day
15      leng = 0
16      value = []
17      date = []
18      while date1 != date2:
19          date1 = datetime.strptime(date1, "%Y-%m-%d")
20          date.append(date1)
21          value.append(rqcsc.selsincsc(f"{date1}", f"{rn}"))
22          print(f"目前進度:{date1}")
23          date1 = datetime.strptime(date1, "%Y-%m-%d")
24          leng += 1
25          date1 += day
26      DataPD = pd.DataFrame(
27          columns=["date", "value"],
28      )
29      for i in range(leng):
30          newrow = pd.DataFrame.from_dict(
31              {"date": [f"{date[i]}"], "value": [f"{value[i]}"]}
32          )
33          DataPD = DataPD.append(newrow, ignore_index=True)
34      return DataPD
35
```

圖十四：取得特定區間資料的程式

在此使用 TimeDelta 將日期調整成可計算，如此一來，便能取得特定區間內的資料。

最後再將此程式包裝成函式，方便後續取用。

```

10 def showSinCSC(date1, date2, rn):
11     day = timedelta(hours=24)
12     d1 = datetime.strptime(date1, "%Y-%m-%d")
13     d2 = datetime.strptime(date2, "%Y-%m-%d")
14     d2 += day
15     leng = 0
16     while d1 != d2:
17         leng += 1
18         d1 += day
19     data = get.getSinCSC(f"{date1}", f"{date2}", f"{rn}")
20     date = []
21     value = []
22     y_index = [i for i in range(0, 105, 5)]
23     for i in range(leng):
24         date.append(str(data["date"][i]))
25         value.append(float(data["value"][i]))
26     plt.style.use("ggplot")
27     plt.figure(figsize=(12, 10), dpi=300)
28     plt.plot(date, value)
29     plt.xlabel("Month", fontsize=10, labelpad=5)
30     plt.ylabel("Percent", fontsize=10, labelpad=10)
31     plt.title(f"{rn} {date1} 至 {date2} 容量趨勢", fontsize=15, y=1)
32     plt.xticks(rotation=45)
33     plt.yticks(y_index)
34     plt.show()

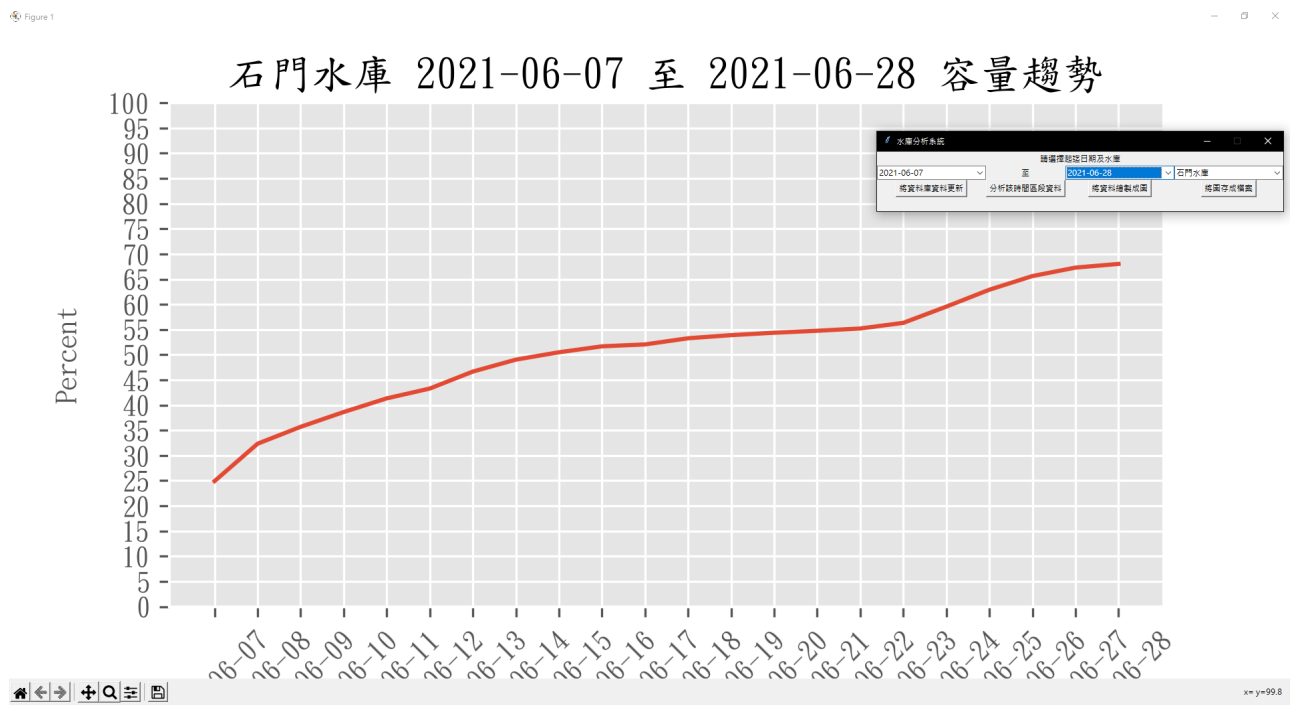
```

圖十五：繪圖

呼叫取得特定區間資料的函式，將其取得資料繪製成圖，最後再將此程式包裝成函式。

方便後續取用。圖為秀出，另有函式為存成圖片。

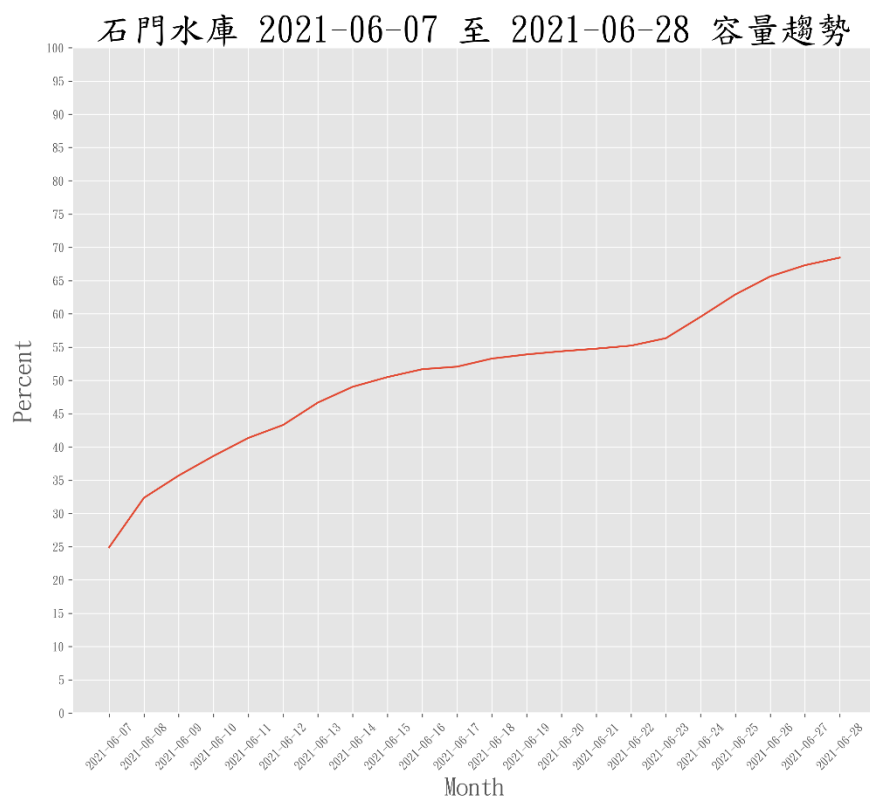
第參章、第六節 GUI



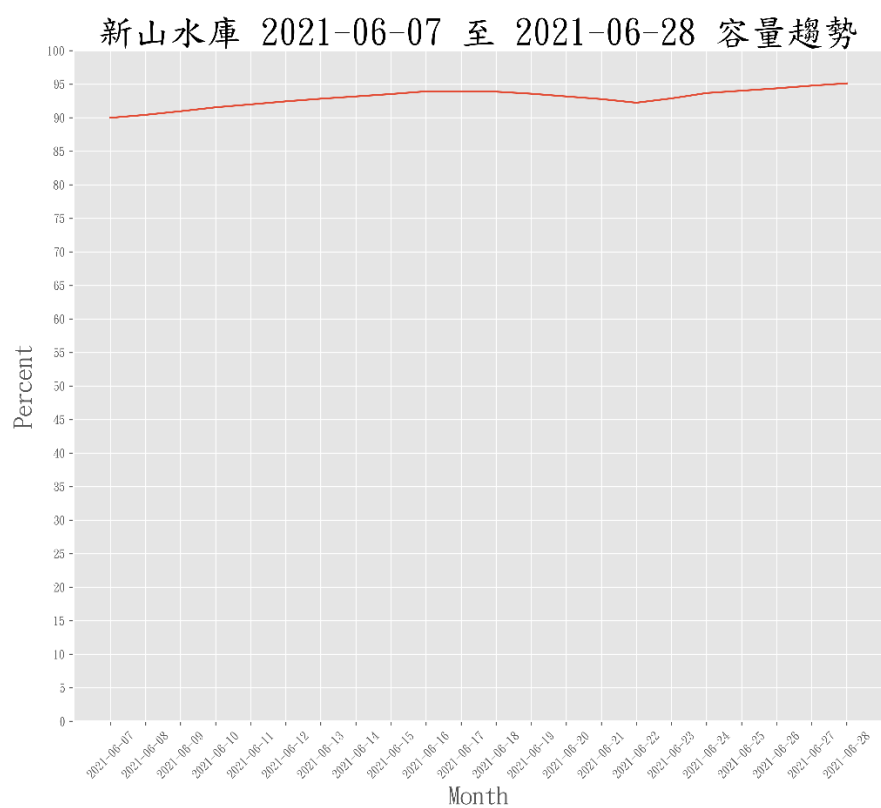
圖十六：GUI 執行畫面

在此使用 Tkinter 繪製 GUI，日期分別對應取得資料時的起迄日期。使得使用者可以更方便取得視覺化後的資料，且可自選區間。且由於 SQL 伺服器公開架設，故在其他電腦上也能執行並取得視覺化後的資料。

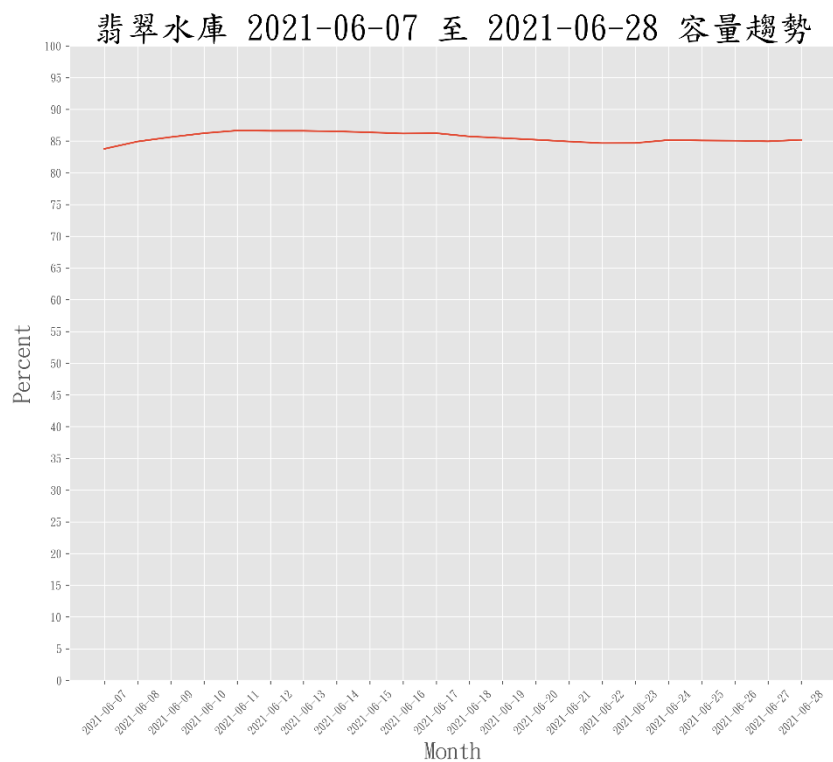
第參章、第七節 結果



圖十七：
石門水庫
2021-06-07
至
2021-06-28
容量趨勢



圖十八：
新山水庫
2021-06-07
至
2021-06-28
容量趨勢



圖十九：

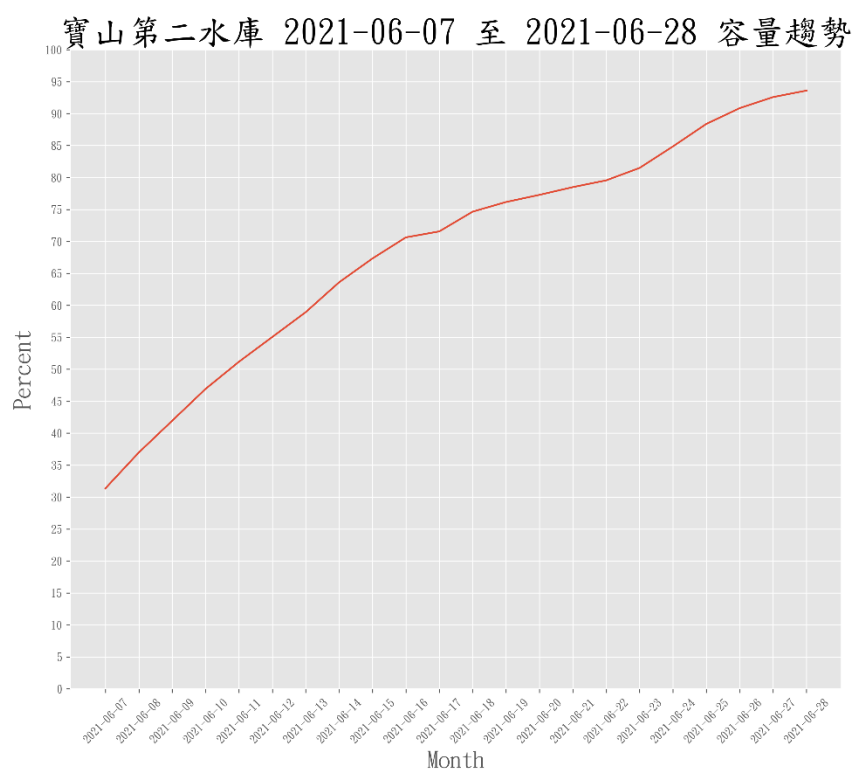
翡翠水庫

2021-06-07

至

2021-06-28

容量趨勢



圖二十：

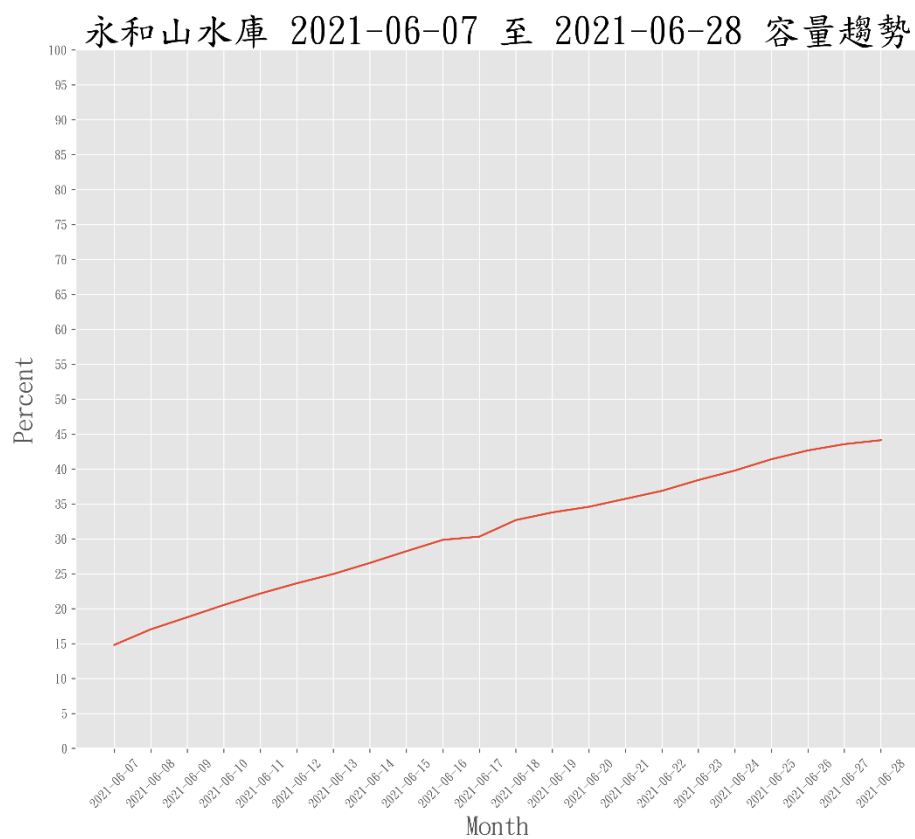
寶山第二水庫

2021-06-07

至

2021-06-28

容量趨勢



圖二十一：

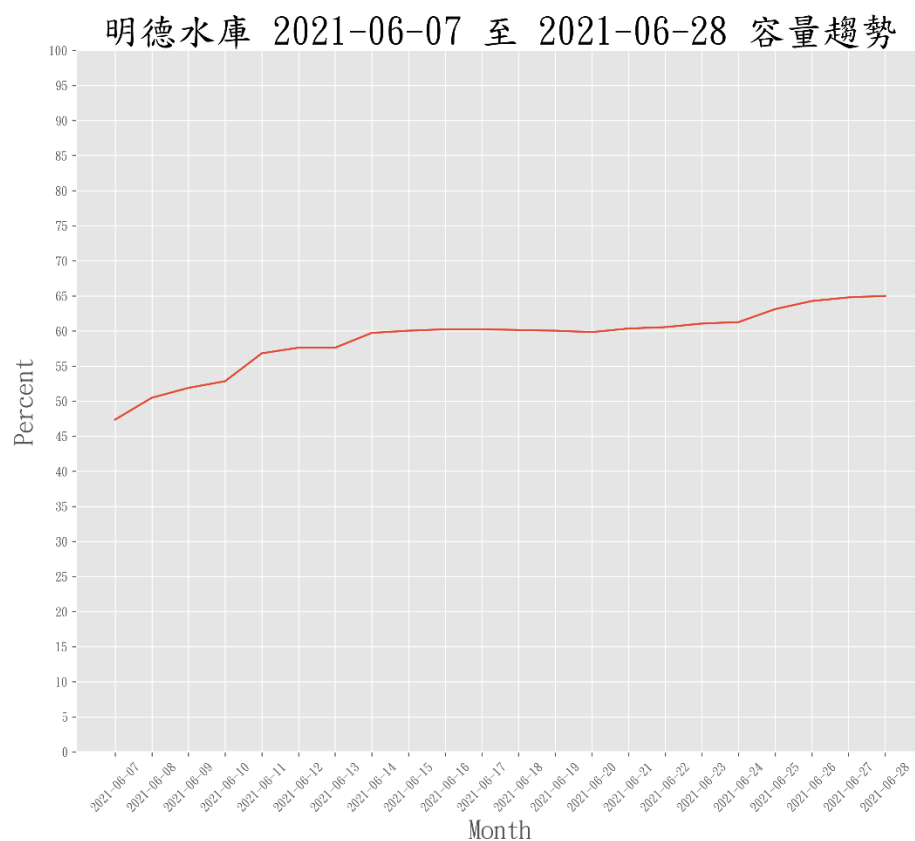
永和山水庫

2021-06-07

至

2021-06-28

容量趨勢



圖二十二：

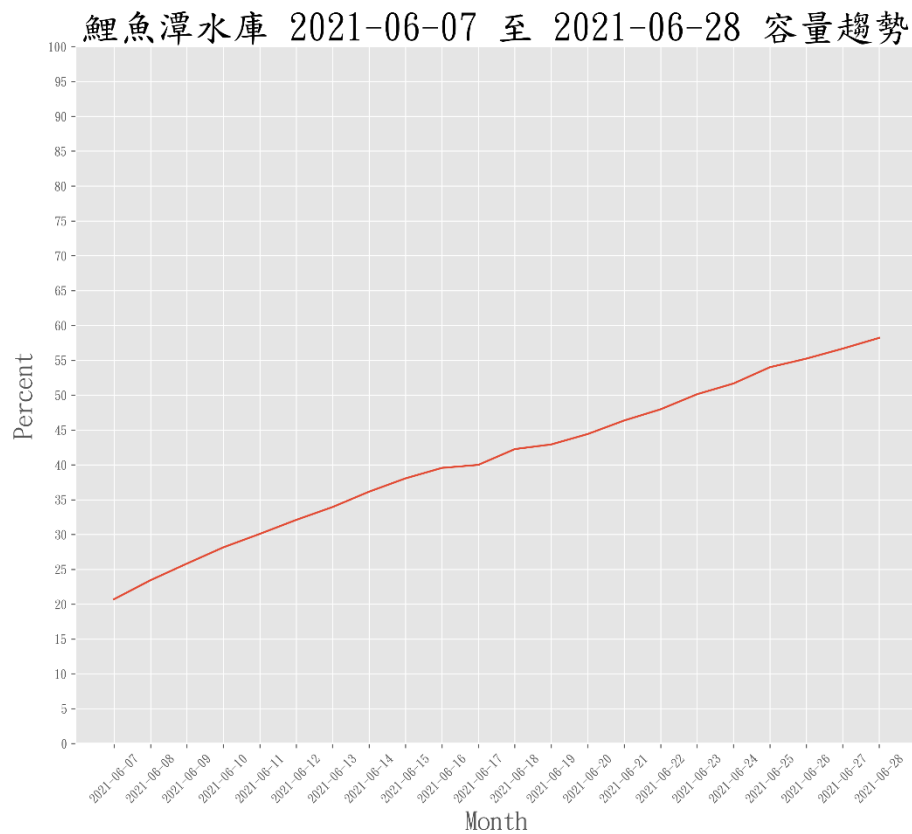
明德水庫

2021-06-07

至

2021-06-28

容量趨勢



圖二十三：

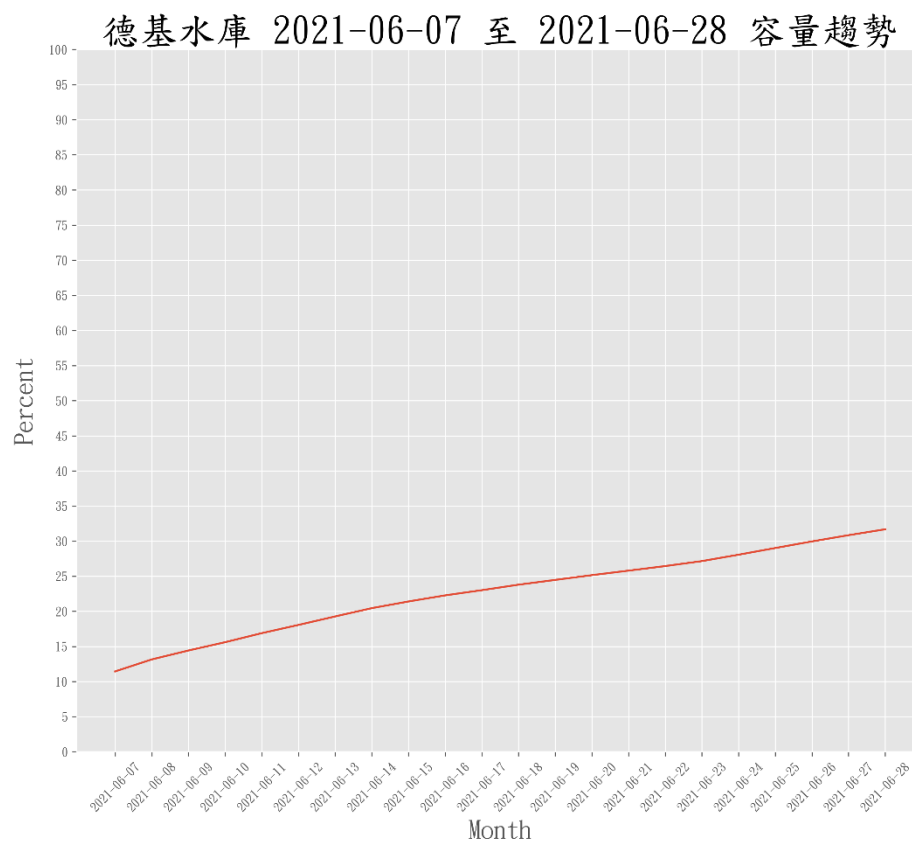
鯉魚潭水庫

2021-06-07

至

2021-06-28

容量趨勢



圖二十四：

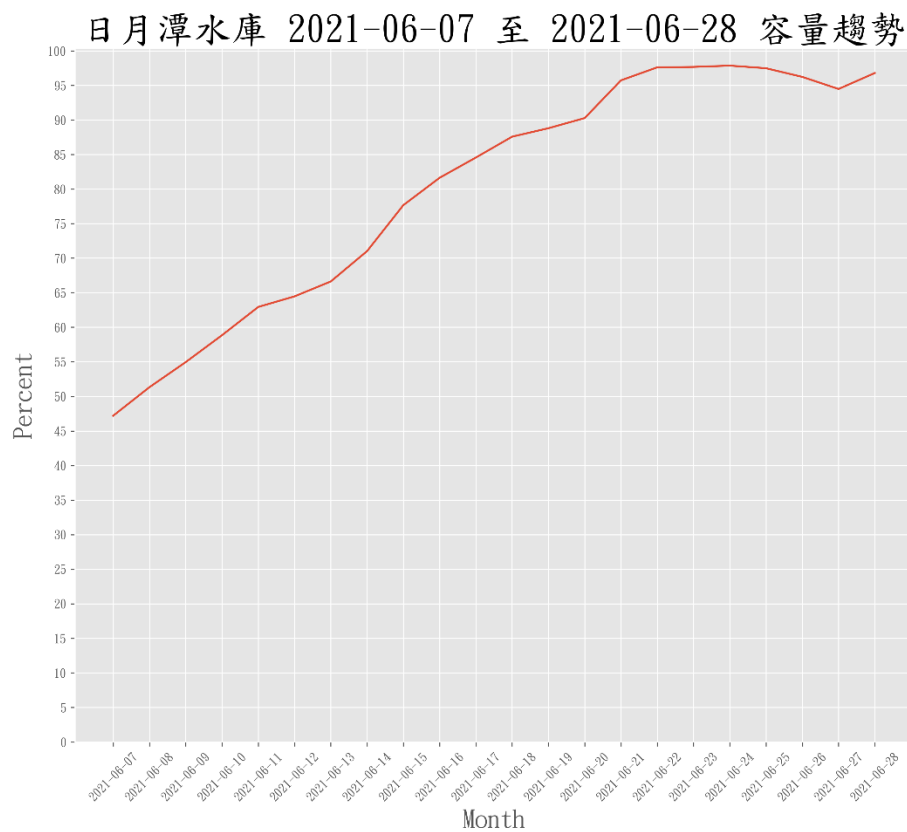
德基水庫

2021-06-07

至

2021-06-28

容量趨勢



圖二十五：

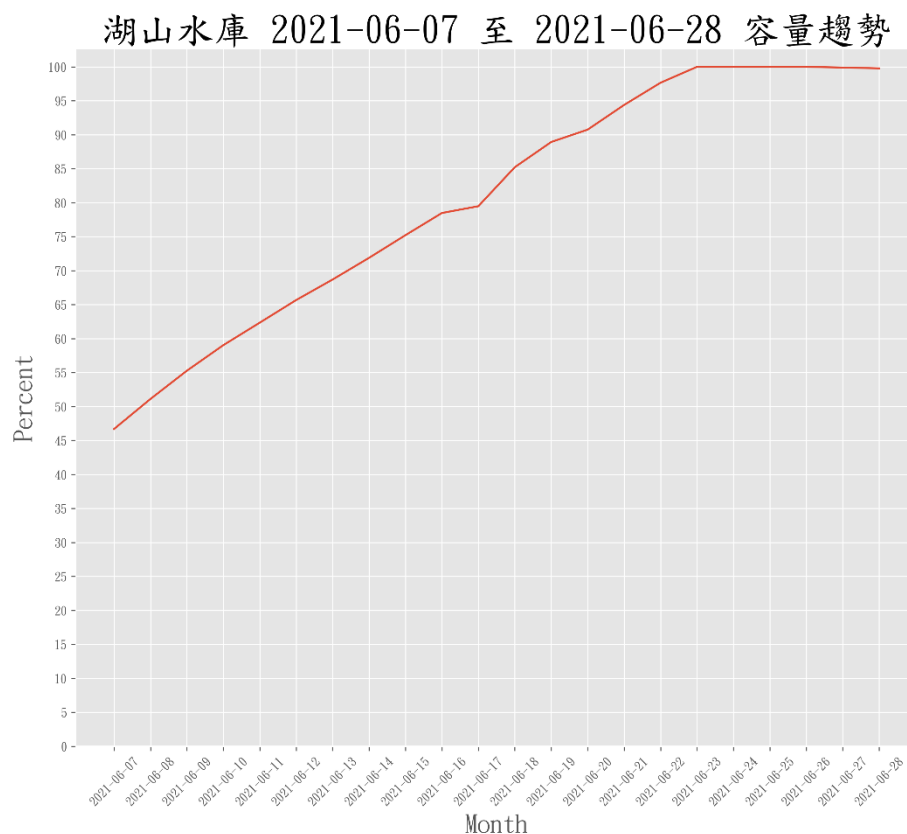
日月潭水庫

2021-06-07

至

2021-06-28

容量趨勢



圖二十六：

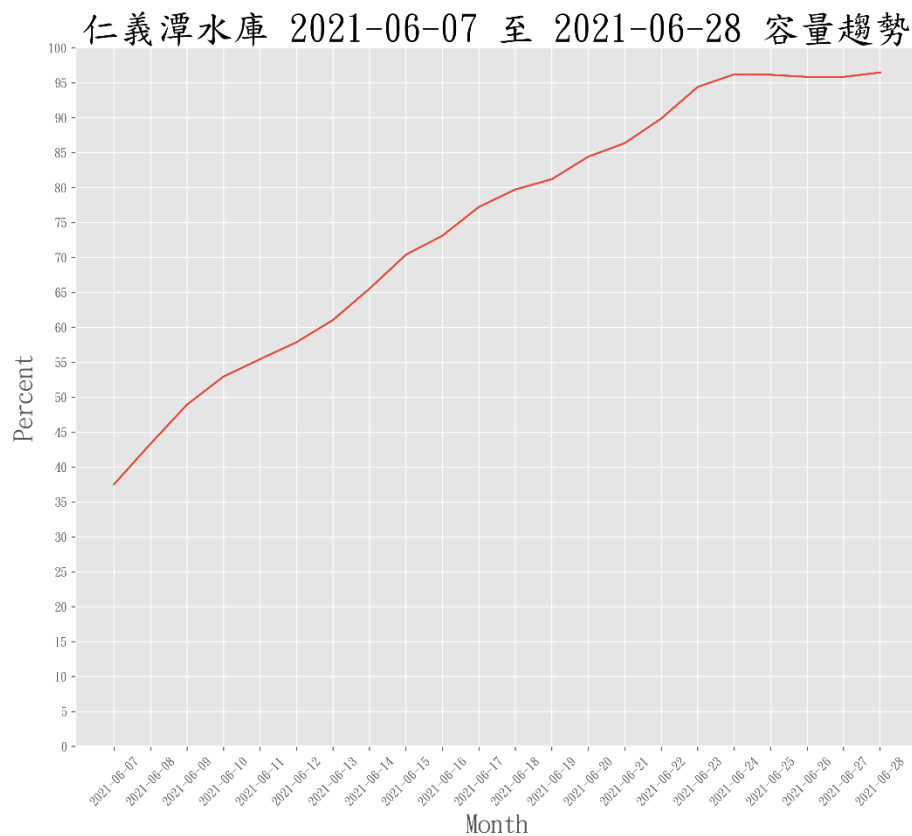
湖山水庫

2021-06-07

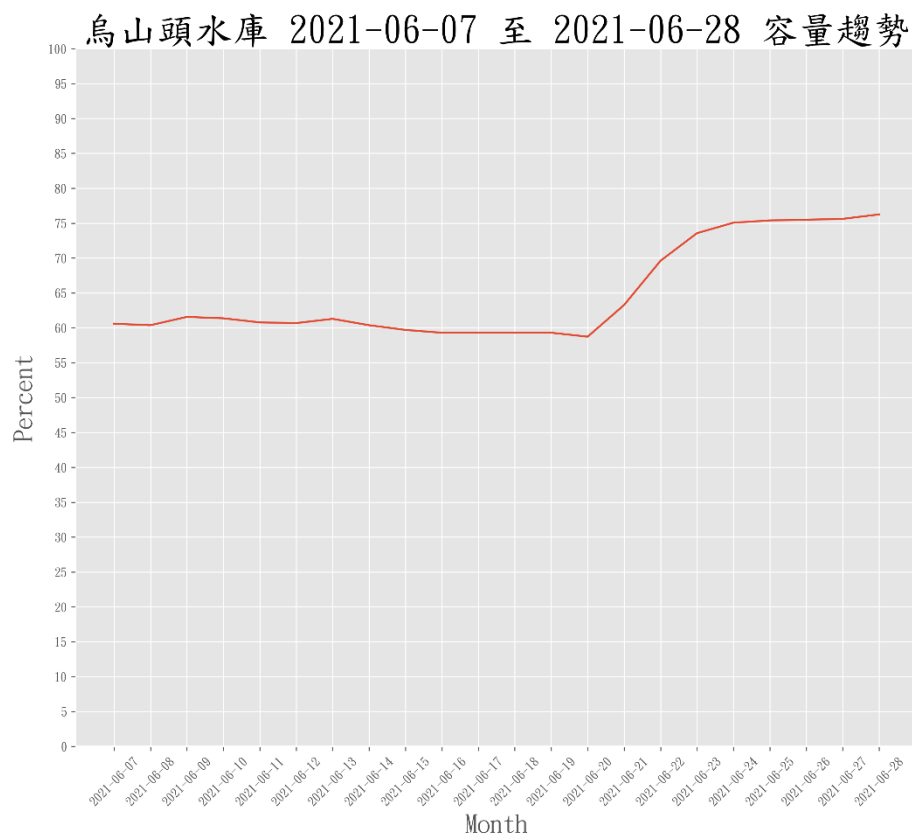
至

2021-06-28

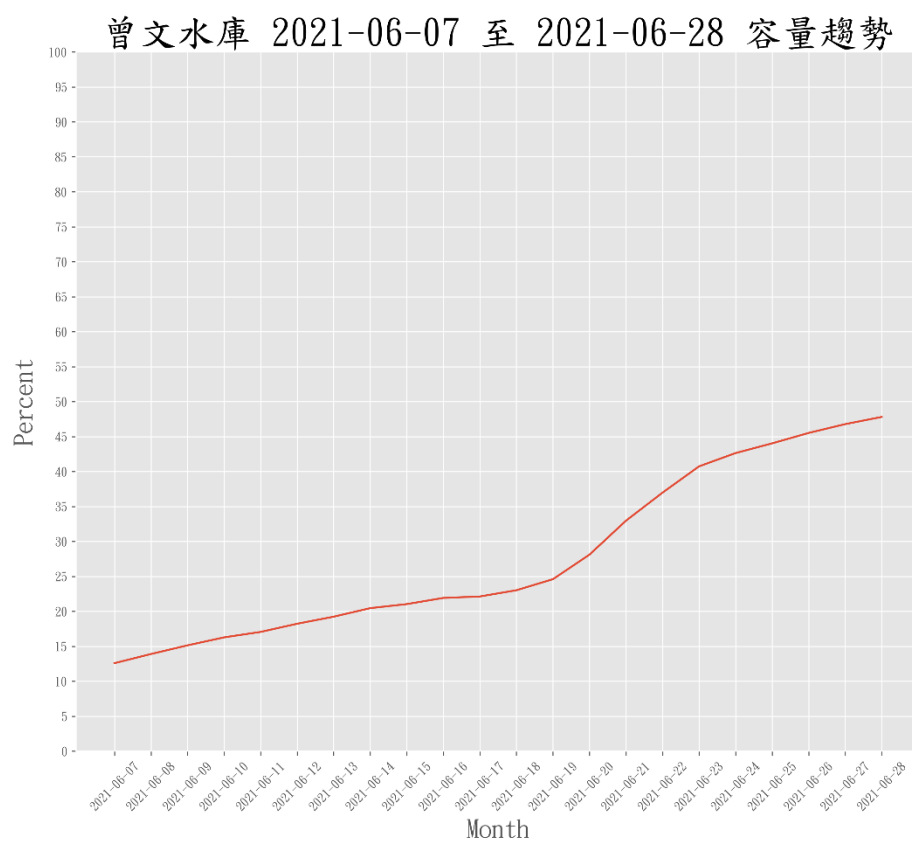
容量趨勢



圖二十七：
仁義潭水庫
2021-06-07
至
2021-06-28
容量趨勢



圖二十八：
烏山頭水庫
2021-06-07
至
2021-06-28
容量趨勢



圖二十九：

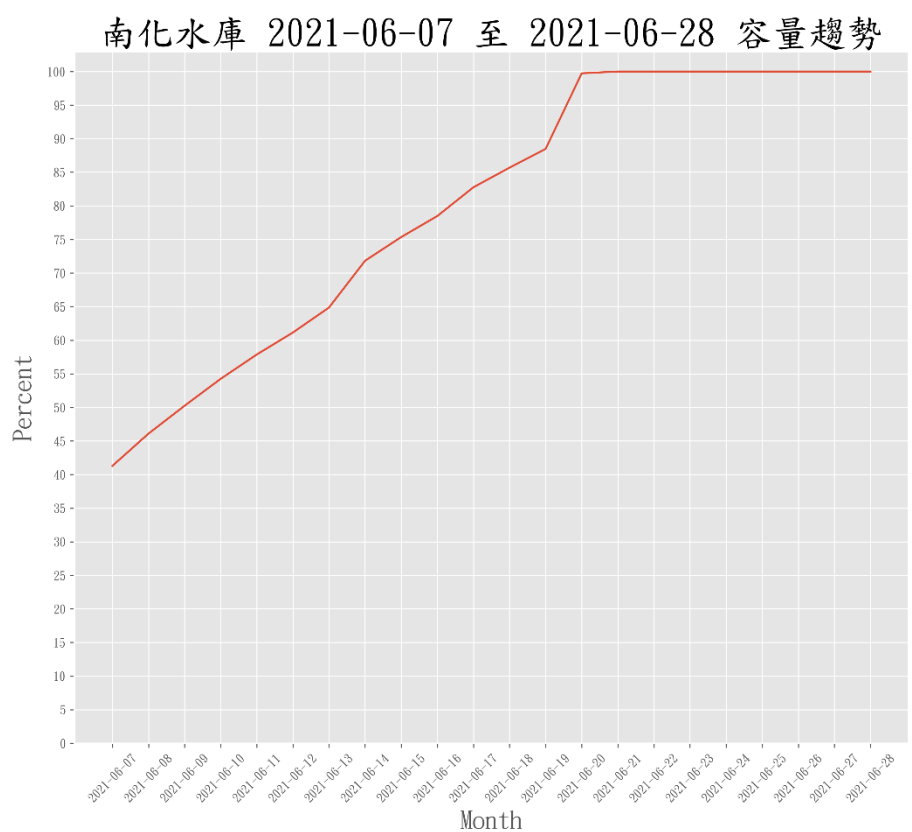
曾文水庫

2021-06-07

至

2021-06-28

容量趨勢



圖三十：

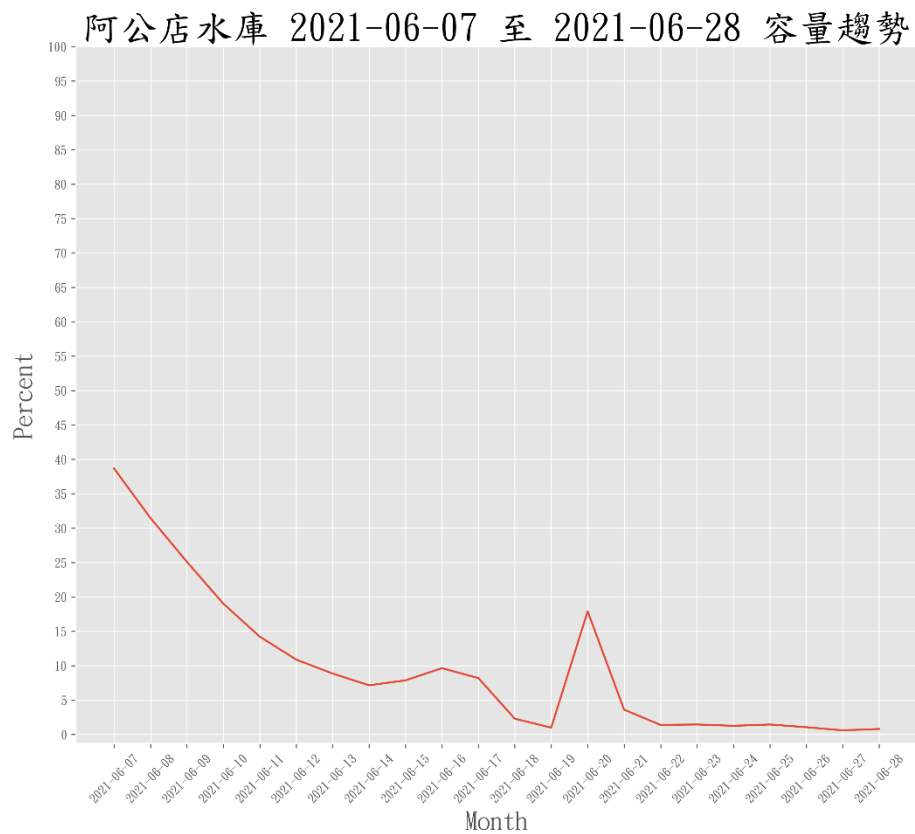
南化水庫

2021-06-07

至

2021-06-28

容量趨勢



圖三十一：

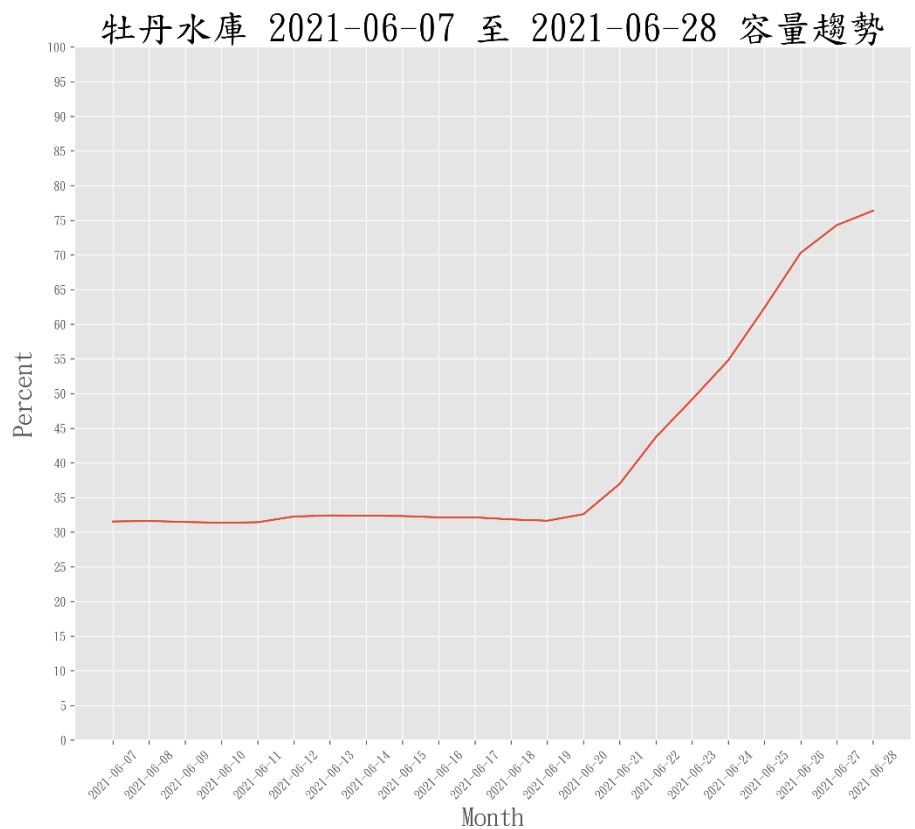
阿公店水庫

2021-06-07

至

2021-06-28

容量趨勢



圖三十二：

牡丹水庫

2021-06-07

至

2021-06-28

容量趨勢

第肆章、結論與未來展望

我認為，這次專題中爬取但沒用到的「有效蓄水量」及「水位高度」沒有到實在有點可惜，也許在未來可以在官方揭露更多消息後來使「水庫衰減量」可以實做出來。而在繪圖的部分，我認為繪製出來的結果還可以更加美觀，甚至是在原有的折線圖上加上其他水庫的資料，來進行比對。而在 GUI 的部分，或許可以新增進度條，使得使用者體驗更佳，而不是在等待時，只能看終端機的輸出或者是乾等。也或許可以把 GUI 取消，並用網頁建置，裝置之間差異等不相容問題或許也會減少。

第伍章、參考文獻

1. Python 使用 BeautifulSoup 抓取與解析網頁資料，開發網路爬蟲教學。【G.T.Wang】。

取自：<https://blog.gtwang.org/programming/python-beautiful-soup-module-scrape-web-pages-tutorial/>

2. datetime --- 基本日期和时间类型。【docs.python.org】。

取自：<https://docs.python.org/zh-cn/3/library/datetime.html>

3. 為應用程式設計圖形化介面，使用 Python Tkinter 模組。【DESIGNSPARK】。

取自：<https://www.rs-online.com/designspark/python-tkinter-cn>

4. 【python 好用模組-matplotlib】好用數據繪圖工具，折線圖、散點圖、長條圖都難不倒

我。【iT 邦幫忙】。取自：<https://ithelp.ithome.com.tw/articles/10232059>

附錄 程式碼

GUI – main_reservoir.py

```
import drawPicture as draw
import reservoir_sel as rqsc
import reservoir_update as up
import getData as get
import tkinter as tk
from tkinter import Button, ttk

print("程式載入中...")
datecombo = rqsc.showtable()
damcombo = rqsc.showrn()

def verificationDate():
    d1 = date1()
    d2 = date2()
    if d1 == d2:
        status = -1
        return status
    elif d1 > d2:
        status = 0
        return status
    else:
        status = 1
        return status

def date1():
    return datecombo[date1_combo.current()]

def date2():
    return datecombo[date2_combo.current()]

def dam():
    return damcombo[dam_combo.current()]

def update():
    mes = up.Update()
    tk.messagebox.showinfo(title=mes, message=mes)

def analysis():
    if verificationDate() == -1:
        tk.messagebox.showwarning(title="警告", message="請勿選擇相同日期!")
    elif verificationDate() == 0:
        tk.messagebox.showwarning(title="警告", message="起日不得晚於迄日!")
    else:
        result_text.set(get.getSinCSC(date1(), date2(), dam()))

def show():
    if verificationDate() == -1:
        tk.messagebox.showwarning(title="警告", message="請勿選擇相同日期!")
    elif verificationDate() == 0:
        tk.messagebox.showwarning(title="警告", message="起日不得晚於迄日!")
    else:
        draw.showSinCSC(date1(), date2(), dam())

def save():
    if verificationDate() == -1:
        tk.messagebox.showwarning(title="警告", message="請勿選擇相同日期!")
    elif verificationDate() == 0:
        tk.messagebox.showwarning(title="警告", message="起日不得晚於迄日!")
    else:
        draw.saveSinCSC(date1(), date2(), dam())
        tk.messagebox.showinfo(
            title="已存", message=f"已將結果存成{dam()}--{date1()}至{date2()}容量趨勢.png"
        )
```

```

win = tk.Tk()
win.title("水庫分析系統")
win.resizable(0, 0)

label1 = tk.Label(win, text="請選擇起迄日期及水庫")
label1.grid(column=0, row=0, columnspan=4)

print("正在取得資料表")
date1_combo = ttk.Combobox(win, values=rqcsc.showtable(), state="readonly")
date1_combo.grid(column=0, row=2)
date1_combo.current(0)

label2 = tk.Label(win, text="至")
label2.grid(column=1, row=2)

date2_combo = ttk.Combobox(win, values=rqcsc.showtable(), state="readonly")
date2_combo.grid(column=2, row=2)
date2_combo.current(0)

print("正在取得資料")
dam_combo = ttk.Combobox(win, values=rqcsc.showrn(), state="readonly")
dam_combo.grid(column=3, row=2)
dam_combo.current(0)

btn_update = Button(text="將資料庫資料更新", command=update)
btn_update.grid(column=0, row=3)

btn_analysis = Button(text="分析該時間區段資料", command=analysis)
btn_analysis.grid(column=1, row=3)

btn_draw = Button(text="將資料繪製成圖", command=show)
btn_draw.grid(column=2, row=3)

btn_save = Button(text="將圖存成檔案", command=save)
btn_save.grid(column=3, row=3)

result_text = tk.StringVar() # 設置動態文字
label_result = tk.Label(win, textvariable=result_text) # 文字
label_result.grid(column=0, row=4, columnspan=4) # 以grid方式放置 文字

win.mainloop()

```

繪圖 – drawPicture.py

```
import getData as get
import matplotlib.pyplot as plt
from datetime import timedelta
from datetime import datetime

font = {"family": "DFKai-SB", "weight": "bold", "size": "12"}
plt.rc("font", **font)

def showSinCSC(date1, date2, rn):
    day = timedelta(hours=24)
    d1 = datetime.strptime(date1, "%Y-%m-%d")
    d2 = datetime.strptime(date2, "%Y-%m-%d")
    d2 += day
    leng = 0
    while d1 != d2:
        leng += 1
        d1 += day
    data = get.getSinCSC(f"{date1}", f"{date2}", f"{rn}")
    date = []
    value = []
    y_index = [i for i in range(0, 105, 5)]
    for i in range(leng):
        date.append(str(data["date"][i]))
        value.append(float(data["value"][i]))
    plt.style.use("ggplot")
    plt.figure(figsize=(12, 10), dpi=300)
    plt.plot(date, value)
    plt.xlabel("Month", fontsize=10, labelpad=5)
    plt.ylabel("Percent", fontsize=10, labelpad=10)
    plt.title(f"{rn} {date1} 至 {date2} 容量趨勢", fontsize=15, y=1)
    plt.xticks(rotation=45)
    plt.yticks(y_index)
    plt.show()

def saveSinCSC(date1, date2, rn):
    day = timedelta(hours=24)
    d1 = datetime.strptime(date1, "%Y-%m-%d")
    d2 = datetime.strptime(date2, "%Y-%m-%d")
    d2 += day
    leng = 0
    while d1 != d2:
        leng += 1
        d1 += day
    data = get.getSinCSC(f"{date1}", f"{date2}", f"{rn}")
    date = []
    value = []
    y_index = [i for i in range(0, 105, 5)]
    for i in range(leng):
        date.append(str(data["date"][i]))
        value.append(float(data["value"][i]))
    plt.style.use("ggplot")
    plt.figure(figsize=(12, 10), dpi=300)
    plt.plot(date, value)
    plt.xlabel("Month", fontsize=20, labelpad=5)
    plt.ylabel("Percent", fontsize=20, labelpad=10)
    plt.title(f"{rn} {date1} 至 {date2} 容量趨勢", fontsize=30, y=1)
    plt.xticks(rotation=45)
    plt.yticks(y_index)
    plt.savefig(f"{rn}-{date1}至{date2}容量趨勢.png")
```



```
import reservoir_sel as rqcsc
from datetime import timedelta
from datetime import datetime
import pandas as pd

day = timedelta(hours=24)
date = datetime.now().strftime("%Y-%m-%d")

def getSinCSC(date1, date2, rn):
    print(f"正在取得 {rn} {date1} 至 {date2} 的資料:")
    date1 = datetime.strptime(date1, "%Y-%m-%d")
    date2 = datetime.strptime(date2, "%Y-%m-%d")
    date2 += day
    leng = 0
    value = []
    date = []
    while date1 != date2:
        date1 = datetime.strptime(date1, "%Y-%m-%d")
        date.append(date1)
        value.append(rqcsc.selsincsc(f"{date1}", f"{rn}"))
        print(f"目前進度:{date1}")
        date1 = datetime.strptime(date1, "%Y-%m-%d")
        leng += 1
        date1 += day
    DataPD = pd.DataFrame(
        columns=["date", "value"],
    )
    for i in range(leng):
        newrow = pd.DataFrame.from_dict(
            {"date": [f"{date[i]}"], "value": [f"{value[i]}"]}
        )
        DataPD = DataPD.append(newrow, ignore_index=True)
    return DataPD
```


SQL 指令 – reservoir_sel.py

```
import pymysql

def selsincsc(date, rn):
    data = []
    conn = pymysql.connect(
        host="xiaoyi1211.ddns.net",
        port=3306,
        user="chy",
        passwd="Henrychy@1211",
        db="reservoir",
        charset="utf8",
    )
    cursor = conn.cursor(pymysql.cursors.DictCursor)
    cursor.execute(f'SELECT `CSC` FROM `{date}` WHERE RN = "{rn}"')
    row = ""
    row = cursor.fetchall()
    rows = row[0]
    value = rows["CSC"]
    data = float(value.replace("%", ""))
    conn.commit()
    cursor.close()
    conn.close()
    return data

def showtable():
    data = []
    conn = pymysql.connect(
        host="xiaoyi1211.ddns.net",
        port=3306,
        user="chy",
        passwd="Henrychy@1211",
        db="reservoir",
        charset="utf8",
    )
    cursor = conn.cursor(pymysql.cursors.DictCursor)
    cursor.execute('SHOW TABLES WHERE Tables_in_reservoir != "dam-data"')
    row = ""
    row = cursor.fetchall()
    for i in row: # 取出value
        for key, value in i.items():
            data.append(value)
    conn.commit()
    cursor.close()
    conn.close()
    return data

def showrn():
    data = []
    conn = pymysql.connect(
        host="xiaoyi1211.ddns.net",
        port=3306,
        user="chy",
        passwd="Henrychy@1211",
        db="reservoir",
        charset="utf8",
    )
    cursor = conn.cursor(pymysql.cursors.DictCursor)
    cursor.execute("SELECT `RN` FROM `dam-data`")
    row = ""
    row = cursor.fetchall()
    for i in row: # 取出value
        for key, value in i.items():
            data.append(value)
    conn.commit()
    cursor.close()
    conn.close()
    return data
```



```
import threading
import reservoir_update as up

def set_interval(func, sec):
    def func_wrapper():
        set_interval(func, sec)
        func()

    t = threading.Timer(sec, func_wrapper)
    t.start()
    return t

set_interval(print(up.Update()), 3600)
```


更新至 SQL – reservoir_update.py

```
import getRequests as RS
import pymysql
from datetime import datetime

def Update():
    date = datetime.now().strftime("%Y-%m-%d") # 時間,Time
    RSN = [] # 水庫名稱, Reservoir name
    ESC = [] # 有效蓄水量, Effective storage capacity
    CWL = [] # 目前水位, Current water level
    CSC = [] # 目前蓄水容量, Current storage capacity
    RT = [] # 記錄時間, Record time

    # 取得所有水庫資料
    data = ["北區", "中區", "南區"]
    for i in data:
        result = RS.reservoir(f"{i}")
        RSN += result[0]
        ESC += result[1]
        CWL += result[2]
        CSC += result[3]
        RT += result[4]

    # 寫入資料庫
    conn = pymysql.connect(
        host="xiaoyi1211.ddns.net",
        port=3306,
        user="chy",
        passwd="Henrychy@1211",
        db="reservoir",
        charset="utf8",
    )
    cursor = conn.cursor(pymysql.cursors.DictCursor)

    # 更新所有資料
    print("正在更新總資料:")
    for i in range(len(RSN)):
        sql = f"UPDATE `dam-data` SET
        `ESC`='{str(ESC[i])}', `CWL`='{str(CWL[i])}', `CSC`='{str(CSC[i])}', `RT`='{str(RT[i])}'
        WHERE `RN`='{str(RSN[i])}'"
        cursor.execute(sql)
        print(f"第{i+1}筆資料")
        conn.commit()

    # 紀錄並更新水位

    # 創建當日資料表
    sql = f"CREATE TABLE IF NOT EXISTS `{date}` (`RN` varchar(20) NOT NULL, `CSC`
    varchar(20) NOT NULL) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;"
    try:
        cursor.execute(sql)
    except Exception as e:
        print("Mysql Error %d: %s" % (e.args[0], e.args[1]))
        pass

    # 更新水位資訊
    sql = f"SELECT * FROM `{date}`"
    cursor.execute(sql)
    if cursor.fetchall() != ():
        print("正在更新水位資料:")
        for i in range(len(RSN)):
            sql = f"UPDATE `{date}` SET `CSC`='{str(CSC[i])}' WHERE `RN` =
            '{str(RSN[i])}'"
            cursor.execute(sql)
            print(f"第{i+1}筆資料")
            conn.commit()
    else:
        print("正在新增水位資料:")
        for i in range(len(RSN)):
            sql = f"INSERT INTO `{date}` (`RN`, `CSC`) VALUES ('{str(RSN[i])}',
            '{str(CSC[i])}')"
            cursor.execute(sql)
            print(f"第{i+1}筆資料")
            conn.commit()
    message = "已更新完成"
    cursor.close()
    conn.close()
    return message
```

```

import requests
from bs4 import BeautifulSoup as BS

def reservoir(area):
    # 爬取資料
    data = {"area": f"{area}"}
    response = requests.get("https://www.wra.gov.tw/common/getReservoir.ashx", data)
    soup = BS(response.text, "html.parser")
    # 區域偵測 Area detection
    currentarea = data["area"]
    print("目前" + f"{currentarea}" + "水庫:")
    # 列出目前水庫資料(有效蓄水量, 目前水位, 目前蓄水容量, 記錄時間)
    damdata = []
    for ul in soup.find_all("ul"):
        for li in ul.find_all("li"):
            damdata.append(li.string)
    # 資料長度
    releng = len(soup.find_all("ul"))
    print("共有" + f"{releng}" + "個水庫")
    dataleng = releng * 4
    # 列出所有搜尋到的水庫
    RSN = []
    span_tags = soup.find_all("span")
    for tag in span_tags:
        RSN.append(tag.string)
    # 擷取有效蓄水量, Effective storage capacity
    ESC = []
    for i in range(0, dataleng, 4):
        ESC.append(str(damdata[i]).replace("有效蓄水量:", "").replace("(萬立方公尺)", ""))
    # 擷取目前水位, Current water level
    CWL = []
    for i in range(1, dataleng, 4):
        CWL.append(str(damdata[i]).replace("目前水位為:", "").replace("(公尺)", ""))
    # 擷取目前蓄水容量, Current storage capacity
    CSC = []
    for i in range(2, dataleng, 4):
        CSC.append(str(damdata[i]).replace("有效蓄水容量:", ""))
    # 擷取記錄時間, Record time
    RT = []
    for i in range(3, dataleng, 4):
        RT.append(str(damdata[i]).replace("記錄時間:", ""))
    # 回傳資料
    result = [RSN, ESC, CWL, CSC, RT]
    return result

def rnleng():
    area = ["北區", "中區", "南區"]
    releng = 0
    for i in area:
        data = {"area": f"{i}"}
        response = requests.get("https://www.wra.gov.tw/common/getReservoir.ashx", data)
        soup = BS(response.text, "html.parser")
        releng += len(soup.find_all("ul"))
    return releng

```