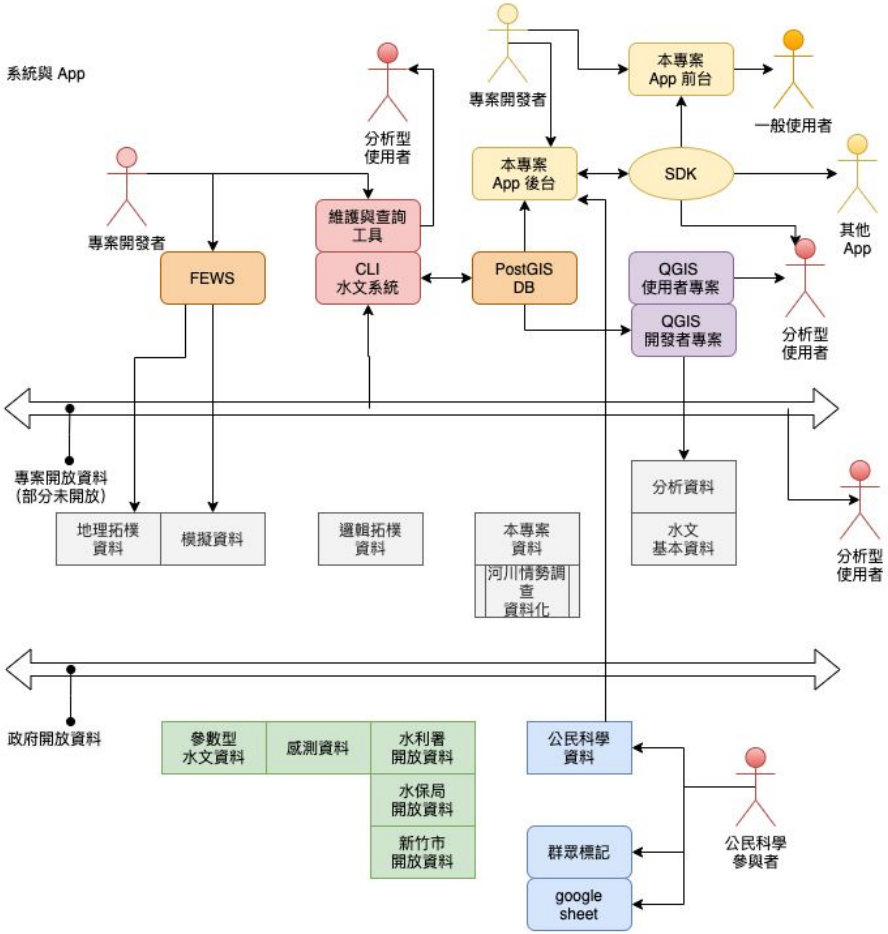
# 水文分析支援包
# V0.3.1

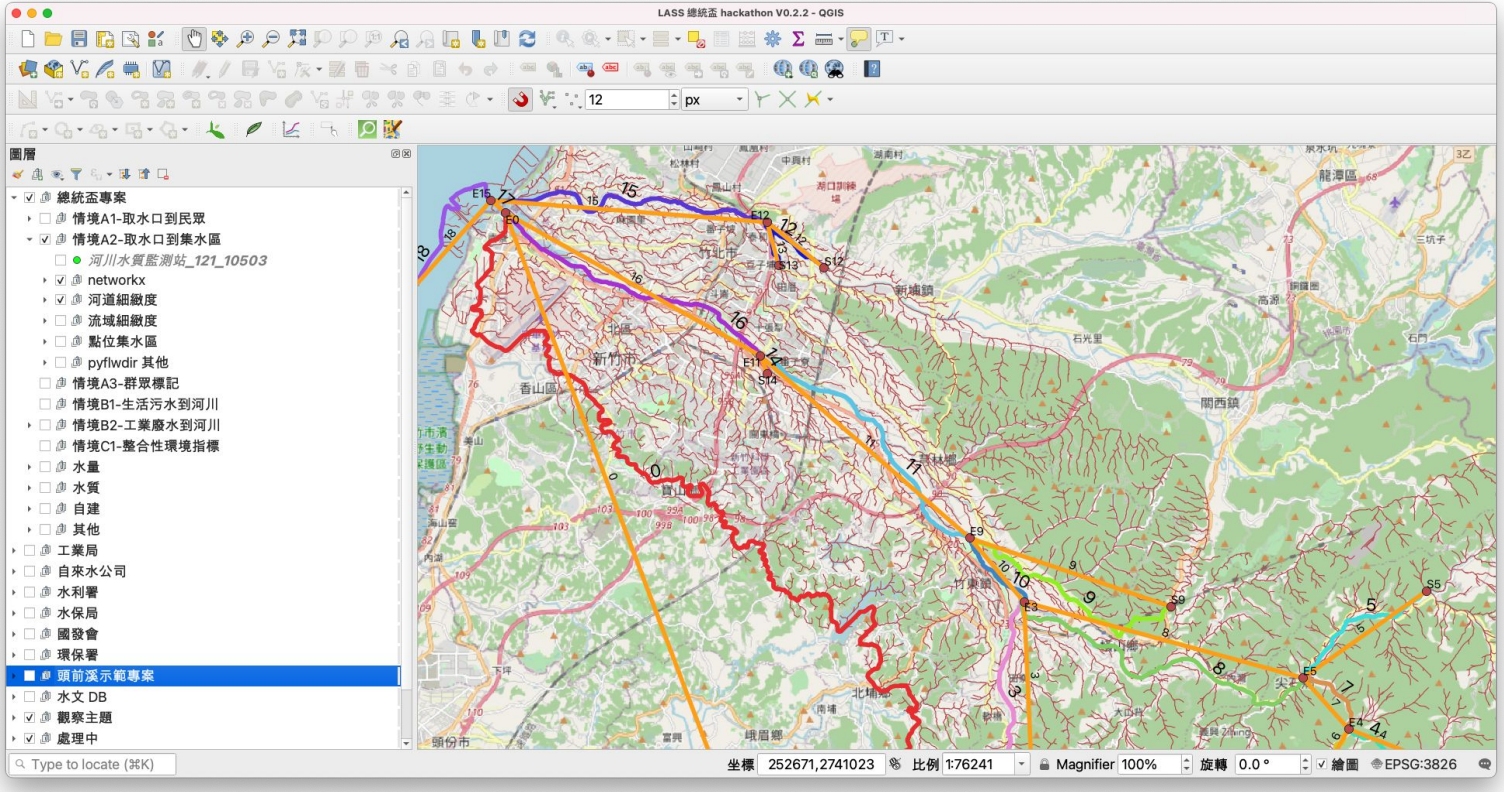
# 2021/08/15 V0.3.1 釋出

- 公開釋出
- 釋出內容
  - QGIS hackathon V0.3.1
    - 文件: QGIS 水開放資料範例專案 - 總統盃黑客松 hackathon
    - 目前圖層列表 QGIS hackathon
      - 92 個圖層含大部分的相關資料
      - 涵蓋基本情境與相關單位
  - 水文分析系統 waterswak V0.3.1
    - 文件: 喝好水 吃好物 有良居-水文分析系統
    - 水文分析系統 CLI V0.3.1 展示: 31:50->41:36
- 分享文
- 喝好水 吃好物 有良居-釋出

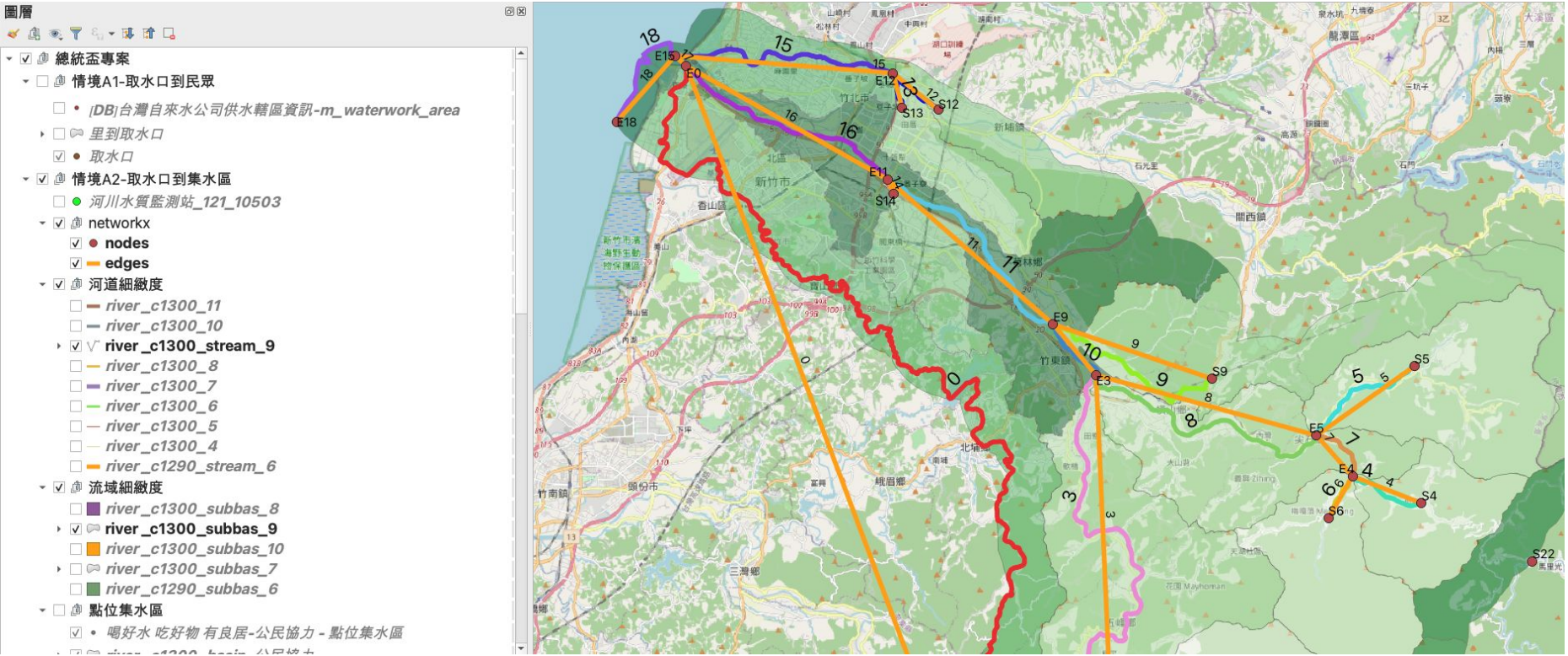
# 系統架構


系統與 App
分析型
使用者
專案開發者
本專案
App 前台
一般使用者
專案開發者
本專案
App 後台
SDK
其他
App
維護與查詢
工具
FEWS
CLI
水文系統
PostGIS
DB
QGIS
使用者專案
QGIS
開發者專案
分析型
使用者
專案開放資料
（部分未開放）
地理拓樸
資料
模擬資料
邏輯拓樸
資料
本專案
資料
河川情勢調查
資料化
分析資料
水文
基本資料
分析型
使用者
政府開放資料
參數型
水文資料
感測資料
水利署
開放資料
水保局
開放資料
新竹市
開放資料
公民科學
資料
群眾標記
google
sheet
公民科學
參與者

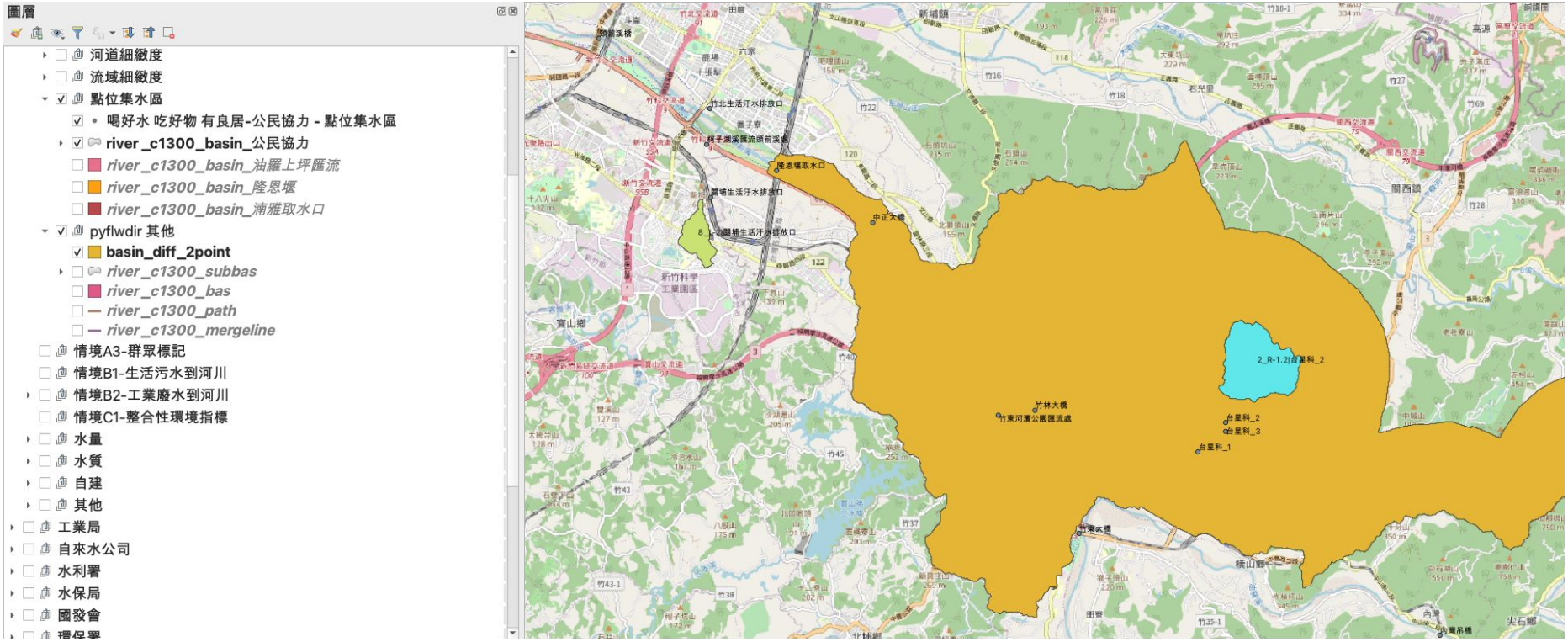
# QGIS hackathon



# 河道流域細緻度


圖層
總統盃專案
情境A1-取水口到民眾
[DB]台灣自來水公司供水轄區資訊-m_waterwork_area
里到取水口
取水口
情境A2-取水口到集水區
河川水質監測站_121_10503
networkx
nodes
edges
河道細緻度
river_c1300_11
river_c1300_10
river_c1300_stream_9
river_c1300_8
river_c1300_7
river_c1300_6
river_c1300_5
river_c1300_4
river_c1290_stream_6
流域細緻度
river_c1300_subbas_8
river_c1300_subbas_9
river_c1300_subbas_10
river_c1300_subbas_7
river_c1290_subbas_6
點位集水區
喝好水 吃好物 有良居-公民協力 - 點位集水區
E15
E0
E18
E12
S12
S13
E11
S14
E9
E3
S9
S5
E5
E4
S4
S6
S22

# 點位集水區/兩點集水區差



# 水文分析系統-CLI

- 結構特性
  - 可以讀 shp, 然後用 sql 查詢, sql 可 auto save to CSV
- sys-系統管理資訊
- basic-水利基本資料
- 流域資訊
- 河川資訊
- 單流域資訊(內建支援頭前溪鳳山溪流域)
  - 河道細緻度(可自行產生)
  - 流域細緻度(可自行產生)
  - 點位集水區
  - 單點找河道最近點
  - 單點產生入海路徑
  - 河道全面支援networkx (邏輯化)
  - 點位查河道最近點
  - 兩點河道距離
- 水文工具
  - shp to CSV
  - 用位置找河川代碼與名稱(外部API)
  - srid 轉換
- 使用維護

目前版本：V0.3.1

```
$ python waterswak.py
root        : INFO     LASS - WaterSWAK version: v0.3.1
WsCLI>help

Documented commands (type help <topic>):
========================================
about  basin      displayall  help  reload_setting  river  tool
basic  catchment  eng         quit  reset           sys

WsCLI>about
WaterSWAK version: v0.3.1
WsCLI>help basin
basin sub command directory
WsCLI>basin
WsCLI:basin>help

Documented commands (type help <topic>):
========================================
desc  help  quit  sql

WsCLI:basin>help desc
desc layer
WsCLI:basin>desc
df key=cli_basin
     gid basin_no basin_name          area  basin_id
extent
0      1     2803       老梅j溪  3.067463e+07    5027.0
```

```
300625.97,2789802.25,307054.59,2799336.00
1      2     2803        阿里磅溪  3.402218e+07    5024.0
304850.50,2790280.75,313167.56,2799191.00
2      3     2803         大坑溪  1.525238e+07    5040.0
298007.91,2791007.50,303860.56,2798297.50
3      4     1090         八連溪  1.555208e+07    3002.0
297281.91,2787596.25,305906.66,2796059.50
4      5     2803        北勢坑溪  1.170417e+07    5046.0
294250.72,2789745.25,302126.31,2795147.75
..   ...      ...         ...           ...       ...
...
138  139     4207        楓港沿海  7.135109e+06    5057.0
```

# 安裝

- 建議使用 anaconda 套件環境
- 先更改 environment.yml 內的 name，改成自己喜歡的環境名稱
- conda env create -f environment.yml

# 使用

- python waterswak.py

# 自訂流域設定

- 需有 DTM, LDD 檔
- 經設定後即可使用

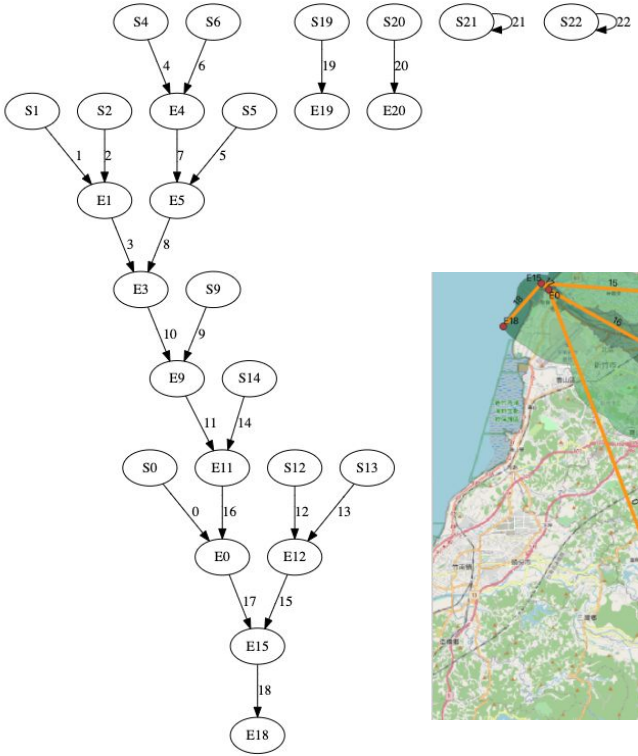
catchment.json

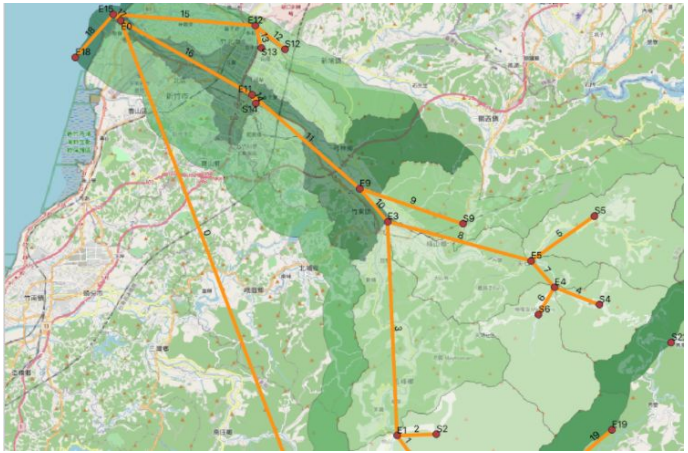
- 每個流域相關資料的設定
  - basin_id 為索引，到 geo 內取得這一筆資料
  - basin_name: 參考用
  - dtm/ldd 是給 pyflwdir 產生河道等資料使用
  - min_sto: 為給 pyflwdir 預設的詳細度設定
  - load_type: 目前只支援 0
- 當有相關 dtm,ldd 資料，可自行按照格式添加，系統即可支援新的流域

```
[
  {
    "basin_id": "1300",
    "basin_name": "頭前溪",
    "geo": "data/basin-河川流域範圍圖/basin-河川流域範圍圖.shp",
    "dtm": "data/catchment/C1300_20m_elv0.tif",
    "ldd": "data/catchment/C1300_20m_ldd.tif",
    "min_sto": 9,
    "load_type": 0
  },
  {
    "basin_id": "1290",
    "basin_name": "鳳山溪",
    "geo": "data/basin-河川流域範圍圖/basin-河川流域範圍圖.shp",
    "dtm": "data/catchment/C1290_20m_elv0.tif",
    "ldd": "data/catchment/C1290_20m_ldd.tif",
    "min_sto": 6,
    "load_type": 0
  }
]
```

# 河道內建 networkx

- 內建 nx, 在 stream 產生時更新
- 結構
  - 將 geo 內勘入 networkx
- Library
  - path, path_length
  - 順向/逆向/不連接
- CLI
  - nx_desc
    - 輸出基本資料， nodes, edges
    - 輸出 json
    - 輸出 dot
  - nx_info
    - path 節點A 節點B
      - 路徑, path_length
      - 順向/逆向/不連接
- 之後
  - output 2point A B
    - 各輸出 point_near
    - 用接在線上的兩點，得知 path, path_l
      - 線中一點到頭尾的距離
    - 真的計算河道距離
- 測試範例
  - 起點在 line index=9, 終點在 line index=16
    - 過程能得知點跟河的距離，最近點為何
    - 過程經過哪些點，哪些線段, 各自距離，總距離



# 了解檢查拓樸

```
WsCLI:catchment>help nx_info
query network information
nx_info 2node node_a node_b
ex: nx_info 2node S0 E18

WsCLI:catchment>nx_info 2node S0 E18
path S0->E18:['S0', 'E0', 'E15', 'E18']
edges S0->E18:[0, 17, 18]
length information:
L0(S0->E0):64969.167845
L17(E0->E15):682.332248
L18(E15->E18):6273.410912
total length=71924.911005
kind S0->E18:1 (1: 順向 -1:逆向 0:沒在
```

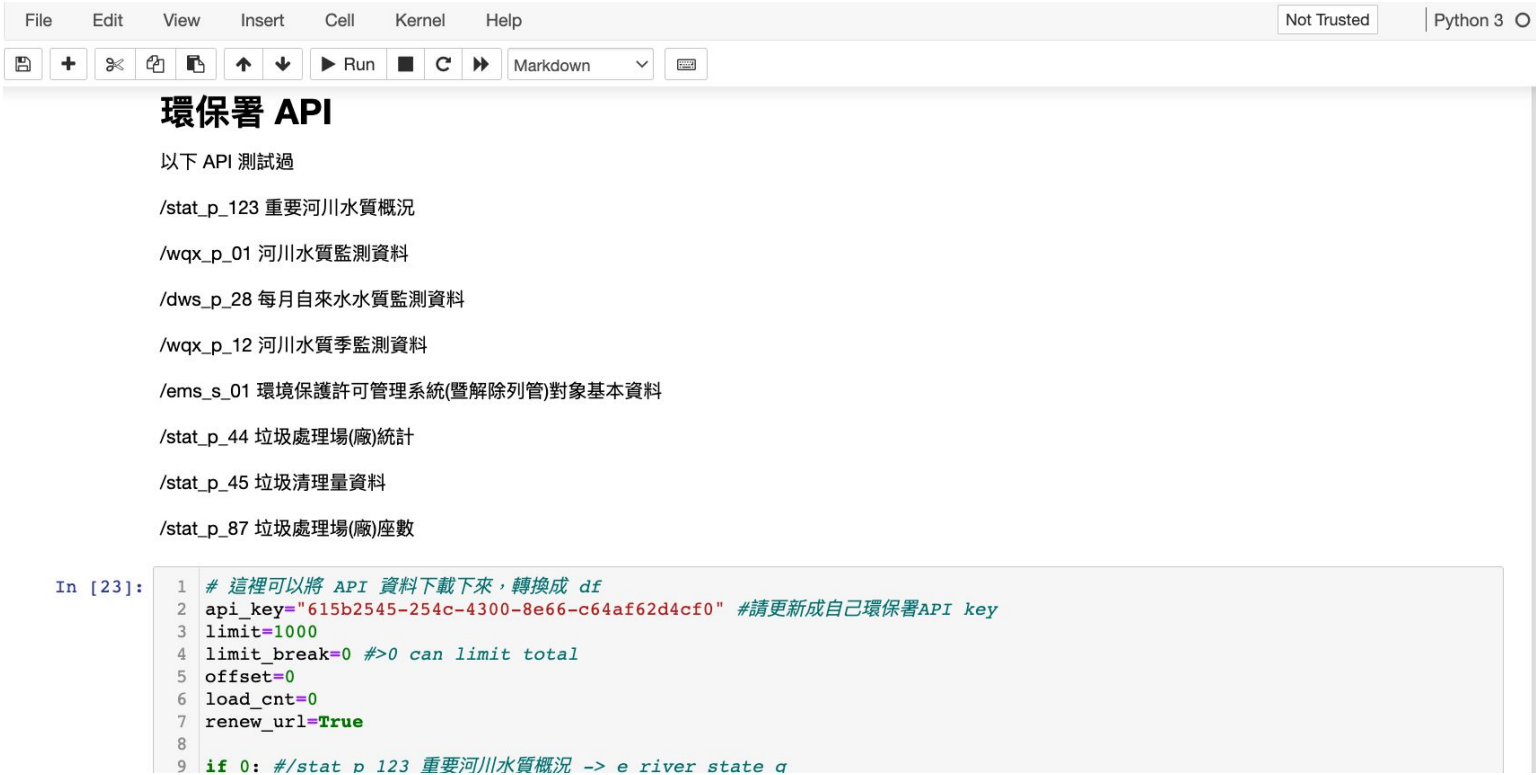
```
WsCLI:catchment>help output
output different level of stream, subbas , point_catchment, downstream path ....
output [type] [...]
    type:
        stream : stream sto from 4-11,
        subbas : sto from 4-11,
        point_catchment_csv csv_filename : -need csv filename input or x,y for single point
        point_catchment x,y : - x,y for single point
        path x,y [x,y] [...] : generate downstream path
        point_near x,y [min_distance] :  get stream nearest information by point, line_index, distance, point_x, point_y
        2point x1,y1 x2,y2 [min_distance] : desc 2point path, path length, river length
        nx_write_shp : output network to shp
ex: output stream
    output subbas
    output point_catchment_csv "data/喝好水 吃好物 有良居-公民協力 - 點位集水區.csv"
    output point_catchment 253520,2743364
    output path 253520,2743364
    output point_near 253520,2743364 5000
    output 2point 262572,2736940 247346,2747132 5000
    output nx_write_shp

WsCLI:catchment>output 2point 262572,2736940 247346,2747132 5000
Point(262572 2736940):
        line_index=9, distance=613.761, point_x=262140.074, point_y=2736503.947
        line_length=10216.604996,length_from_start=6038.236745
Point(247346 2747132):
        line_index=16, distance=376.902, point_x=247158.355, point_y=2746805.129
        line_length=11277.803220,length_from_start=5994.800305
path sequence:['S9', 'A', 'E9', 'E11', 'B', 'E0']
edges :[9, 11, 16]
L11(E9->E11):11352.370215
river length between A(262572,2736940),B(247346,2747132): 21525.538771
```

# python notebook

- engineer_mode.ipynb
  - 拓墣基礎與使用
  - 拓墣相關演算法練習
  - 測站在河流的順序與距離
  - 分析淨水場水質 from DB
  - plot 支援
  - 山河事件簿相關
  - Grafana 連結
  - 情境
    - 情境 - A1
  - flwdir
    - 相關基礎測試 geopandas, shapely
    - 帶起各種 flwdir 功能
    - 站點到 stream 的距離
    - flwdir+networkx
  - 工業局API
  - 河川代碼查詢
  - 在地存取地理資料
- maintain.ipynb
  - shp 匯入資料庫
  - 淨水場水質灌入資料庫與分析
  - 環保署 API 支援與灌入資料庫
    - EPA API 關於專案有關心的部分，大致都有進入資料庫
  - EPA CWMS 水量水質自動監測連線處理
  - 將 panda dataframe 直接建 DB 內資料表與灌入資料
  - Data Transfer Kit

# engineer_mode.ipynb



頭前溪範圍一公里內水位測站
dist: 與河流距離,loc:在頭前溪河流(總長 71235.71 m)的位置比例
loc_dist_m:與下一個測站河流距離 (m) ,loc_time_h: 以平均流速 12000.00 m/h, 需要多久(h)
point_dist_m: 與下一個測站直線距離(m),curve_rate: 河流距離/直線距離的比例
height_m: 測站高程(m),slope: 測站間平均坡度=距離/高度差

Out[4]:

| | type | id | name | dist | loc | geom_wkt | loc_dist_m | loc_time_h | point_dist_m | curve_rate | height_m | slope |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 河川水位測站 | 1300H020 | 五峰大橋 | 161.86 | 0.45 | POINT(262524.23 2726396.01) | 3,912.76 | 0.33 | 2,891.98 | 1.35 | 262.00 | 78.26 |
| 4 | 河川水位測站 | 1300H014 | 上坪 | 41.19 | 0.50 | POINT(261568.14 2729125.38) | 12,769.63 | 1.06 | 8,957.34 | 1.43 | 212.00 | 116.09 |
| 10 | 河川水位測站 | 1300H016 | 竹林大橋 | 228.14 | 0.68 | POINT(259465.76 2737832.5) | 6,392.21 | 0.53 | 5,598.77 | 1.14 | 102.00 | 152.20 |
| 12 | 河川水位測站 | 1300H023 | 中正大橋 | 102.83 | 0.77 | POINT(255763 2742032) | 4,740.64 | 0.40 | 4,037.23 | 1.17 | 60.00 | 182.33 |
| 14 | 河川水位測站 | 1300H024 | 柯子湖溪匯流站 | 264.71 | 0.84 | POINT(252144.44 2743822.32) | 526.47 | 0.04 | 698.67 | 0.75 | 34.00 | 87.74 |
| 15 | 河川水位測站 | 1300H017 | 經國橋 | 132.46 | 0.85 | POINT(251968.79 2744498.55) | 8,923.80 | 0.74 | 7,973.18 | 1.12 | 28.00 | 405.63 |
| 21 | 河川水位測站 | 1300H021 | 舊港橋 | 307.98 | 0.97 | POINT(244886.69 2748161.26) | 1,449.10 | 0.12 | 1,345.90 | 1.08 | 6.00 | 289.82 |
| 25 | 河川水位測站 | 1300H022 | 竹港大橋 | 0.08 | 0.99 | POINT(243658.05 2748710.71) | 644.76 | 0.05 | 644.76 | 1.00 | 1.00 | 644.76 |

# 分析淨水場水質 from DB

In [63]:

```
1 item='總溶解固體量(Total Dissolved Solids)'
2 waterwork="新竹第二淨水場"
```

# 其他工具(安裝環境內)

- 簡單說明與範例請參考水文分析系統文件
- rasterio/rio CLI (網格工具)
- Fiona/fio CLI(向量工具)

```
$ rio --help
Usage: rio [OPTIONS] COMMAND [ARGS]...

  Rasterio command line interface.

Options:
  -v, --verbose           Increase verbosity.
  -q, --quiet             Decrease verbosity.
  --aws-profile TEXT      Select a profile from the AWS credentials file
  --aws-no-sign-requests  Make requests anonymously
  --aws-requester-pays    Requester pays data transfer costs
  --version               Show the version and exit.
  --gdal-version
  --help                  Show this message and exit.

Commands:
  blocks     Write dataset blocks as GeoJSON features
  bounds     Write bounding boxes to stdout as GeoJSO
  calc       Raster data calculator.
  clip       Clip a raster to given bounds.
  convert    Copy and convert raster dataset.
  edit-info  Edit dataset metadata.
  env        Print information about the Rasterio env
  gcps       Print ground control points as GeoJSON.
  info       Print information about a data file.
  insp       Open a data file and start an interprete
  mask       Mask in raster using features.
  merge      Merge a stack of raster datasets.
  overview   Construct overviews in an existing datas
  rasterize  Rasterize features.
  rm         Delete a dataset.
  sample     Sample a dataset.
  shapes     Write shapes extracted from bands or mas
  stack      Stack a number of bands into a multiband
  transform  Transform coordinates.
  warp       Warp a raster dataset.
```

```
Usage: fio [OPTIONS] COMMAND [ARGS]...

  Fiona command line interface.

Options:
  -v, --verbose     Increase verbosity.
  -q, --quiet       Decrease verbosity.
  --version         Show the version and exit.
  --gdal-version    Show the version and exit.
  --python-version  Show the version and exit.
  --help            Show this message and exit.

Commands:
  bounds   Print the extent of GeoJSON objects
  calc     Calculate GeoJSON property by Python expression
  cat      Concatenate and print the features of datasets
  collect  Collect a sequence of features.
  distrib  Distribute features from a collection.
  dump     Dump a dataset to GeoJSON.
  env      Print information about the fio environment.
  filter   Filter GeoJSON features by python expression.
  info     Print information about a dataset.
  insp     Open a dataset and start an interpreter.
  load     Load GeoJSON to a dataset in another format.
  ls       List layers in a datasource.
  rm       Remove a datasource or an individual layer.
```

# 研究筆記章節

- 測站總表-sensor_station
- 如何排測站在流域內的上下游順序
- 測站在河流的順序與距離
- 河川代碼查詢
- 點位集水區-公民協力
- rasterio/rio 學習 / Fiona/fio 學習