水資源物聯網

Web Services 使用手冊

V1.2 Draft

本手冊所提及的軟體是在授權合約之下提供，只有在同意以下使用授權合約的情況下方可使用本手冊中介紹的產品。

**版權公告**

Copyright ©2018 AnaSystem Inc.

All rights reserved.

未經安研科技股份有限公司書面允許，任何公司或個人不得以任何形式複製本手冊全部或部分內容。

**商標**

Senslink是AnaSystem Inc.的註冊商標

本手冊中的其它商標分別屬於各自的公司所有。

**重要聲明**

本手冊中提供的資訊並不代表安研科技股份有限公司對這些資訊提供了相應的保證。除非另有約定，本手冊僅作為使用指導，本手冊中的所有陳述、資訊和建議不構成任何明示或暗示的擔保。安研科技股份有限公司對保留隨時對此手冊及手冊中所提及的產品或程式設計改善或修改的權利。

安研科技股份有限公司對努力使本手冊提供的資訊準確和適用，然而安研科技股份有限公司對並不對這些資訊的使用承擔任何責任或任何連帶責任。

產品及使用手冊可能包含技術或印刷上的錯誤。安研科技股份有限公司對將定期更正這些錯誤，並於整合錯誤更新之後發行新版文件。

版本說明

|  |  |
| --- | --- |
| 0.1 | 2016/9/23 |
|  | 初始草稿 |
| 0.2 | 2016/10/8 |
|  | 1. API修改為 REST URI格式 |
| 0.3 | 2016/10/14 |
|  | 1. 認證使用OAuth 2.0 2. 修正API URI |
| 0.4 | 2016/11/28 |
|  | 1. 修改API URI，以物件為根目錄，例如GetStations改為Station/Get，以利後續發展Set, Create等方法 |
| 0.5 | 2017/2/5 |
|  | 1. 增加二維等高線圖API |
| 0.6 | 2017/7/10 增加同期比較統計運算 |
| 1.0 | 2017/9/6 增加 R 連接範例 |
| 1.1 | 2018/9/12 增加Chart功能 |
| 1.2 | 2018/9/26 增加Spark功能 |

**目錄**

[1. 簡介 5](#_Toc528842894)

[2. 名詞定義 5](#_Toc528842895)

[3. 物件結構 5](#_Toc528842896)

[4. Web Servcie 連線位址 6](#_Toc528842897)

[4.1 連線根目錄位址 6](#_Toc528842898)

[5. 安全性驗證方法 6](#_Toc528842899)

[5.1 驗證標準 6](#_Toc528842900)

[5.2 驗證機制 7](#_Toc528842901)

[5.3 驗證伺服器及路徑 7](#_Toc528842902)

[5.4 註冊第三方應用程式Client使用資訊 7](#_Toc528842903)

[5.5 Web Service 存取權限範圍 7](#_Toc528842904)

[5.6 Client Credentials驗證範例 7](#_Toc528842905)

[6. Web Service API 功能 10](#_Toc528842906)

[6.1 命名規則 10](#_Toc528842907)

[6.2 輸入參數與回傳型態使用注意事項 10](#_Toc528842908)

[6.3 子物件取用說明 11](#_Toc528842909)

[6.4 REST API 功能及路徑總表 12](#_Toc528842910)

[6.5 監測站群組 (ETL API) 15](#_Toc528842911)

[6.6 監測站 (ETL API) 15](#_Toc528842912)

[6.7 物理量 (ETL API) 16](#_Toc528842913)

[6.8 物理量資料 (ETL API) 16](#_Toc528842914)

[6.9 圖表 (可視化API) 18](#_Toc528842915)

[6.10 自定分散式運算 (雲端運算API) 20](#_Toc528842916)

[6.11 定型分散式運算 (大數據運算API) 21](#_Toc528842917)

[6.12 點陣地理圖資 22](#_Toc528842918)

[7. 資料結構定義 24](#_Toc528842919)

[7.1 列舉型態 25](#_Toc528842920)

[8. 使用 R 讀取 水資源物聯網 Web Service 範例 25](#_Toc528842921)

[8.1 一般說明 25](#_Toc528842922)

[8.2 所需套件 25](#_Toc528842923)

[8.3 取得Token 25](#_Toc528842924)

[8.4 讀取彙總歷史資料 25](#_Toc528842925)

[9. 使用Python讀取 水資源物聯網 Web Service 範例 26](#_Toc528842926)

[9.1 所需套件 26](#_Toc528842927)

[9.2 取得Token 26](#_Toc528842928)

[9.3 讀取所有監測站資訊 27](#_Toc528842929)

* 1. 簡介

本文件說明如何使用Web Service讀取水資源物聯網所接收與儲存的資料，使用前必須詳細瞭解水資源物聯網資料儲存結構，方能順利取得資料。

* 1. 名詞定義

本節說明Web Service中所使用英文定義與中文對照說明，括弧內為縮寫。

* + - * 1. Alarm: 警戒值，指某個監測物理量達到危險或警告程度的設定值。
        2. Physical Quantity (PQ): 監測物理量，如水位、流量、雨量等。
        3. Physical Quantity ID (PqId): 每個監測物理量，或稱為監測項目，於水資源物聯網系統中都有一個獨一無二的ID，此ID用於讀取歷史資料，取得警戒，讀取歷史資料等功能。
        4. RdqDevice (RD): Remote Data Acquisition Device監測資料來源，可能是一台記錄器，或是第三方資料庫。
        5. Station (ST): 監測站，如雨量站、水位站等，每個監測站有一至數個監測物理量。
        6. Real Time Data: 即時資料，記錄器或其他資料來源傳回的即時資料，與歷史資料有所區隔。
        7. Historical Data: 歷史資料，儲存於水資源物聯網資料庫中的某監測物理量過去所有的歷史資料。
        8. Aggregate Method: 彙總運算方法，針對一段時間的原始歷史資料，進行彙總運算的方法，0: 回傳原始資料，不進行任何計算，1: 將原始資料進行每個小時的平均或累計後再回傳，2: 將原始資料進行每日的平均或累計後再回傳。
  1. 物件結構

對於使用者來說，監測站是監測系統的主體，每一個監測站中包含數個監測物理量(Physical Quantity)，但實際上，每個監測站中監測物理量可能是來自於相同、或不同的資料來源(Remote Data Acquisition Device, RD)，例如，某個實體監測站，安裝了雨量計、水位計、水質儀器，分別取的雨量、水位、酸鹼度數種觀測值，但在管理上，卻分屬於水位站、雨量站、水質站三種監測站，因此實體監測站可視為一個資料來源，而邏輯上的水位站、雨量站、水質站卻分屬邏輯或管理上的三個監測站。(範例詳圖1 PQ M3)

本Web Service中所稱的監測站(Station)，均是以邏輯或管理上的監測站為定義，以便於第三方開發軟體時，與最終使用者的認知一致。水資源物聯網操作畫面上的監測站，也是邏輯監測站。

下圖顯示監測站ST1，擁有M1, M2監測物理量，監測站ST2擁有M3監測物理量，監測站ST3擁有M3, M4, M5監測物理量，監測站ST4擁有M6監測物理量。



1. 資料來源、監測物理量與監測站關聯
   1. Web Servcie 連線位址
      1. 連線根目錄位址

水資源物聯網之web service連線根目錄(root)為https://iapi.wra.gov.tw，通訊規格符合REST，點選該連線亦可取得所有web service的命令參數與回應內容。

Web Service線上說明可參考https://{root}/v3/api/swagger

* 1. 安全性驗證方法
     1. 驗證標準

OAuth 2.0 <https://oauth.net/2/>

* + 1. 驗證機制

OAuth 2.0 驗證機制共有四種，本系統實作Client Credentials及Authorization Code兩種。

* + 1. 驗證伺服器及路徑

連線至<https://{root}/v3/oauth2>，使用水資源物聯網帳號、密碼進行驗證，驗證成功取得Token後，使用此Token進行REST API操作。

其中 Authorization Code驗證法的驗證網頁，路徑為

https://{root}/v3/oauth2/Authorize

Token取得路徑 https://{root}/v3/oauth2/token

* + 1. 註冊第三方應用程式Client使用資訊
       1. 開發者帳號

欲使用Web Service開發第三方應用程式，讀取水資源物聯網資料，需先至水資源物聯網網站註冊開發者帳號，水資源物聯網之開發者帳號與使用者帳號是相同的，因此若您已經有水資源物聯網使用者帳號，則不須再行註冊。

* + - 1. 建立應用程式Client Id與Client Secret

進入水資源物聯網網站，於第三方應用程式管理頁面，為應用程式建立一組新的Client Id與Client Secret，作為之後應用程式取的Token的依據。

若要中止應用程式與水資源物聯網的連結，與此網頁將Client Id與Client Secret刪除即可。

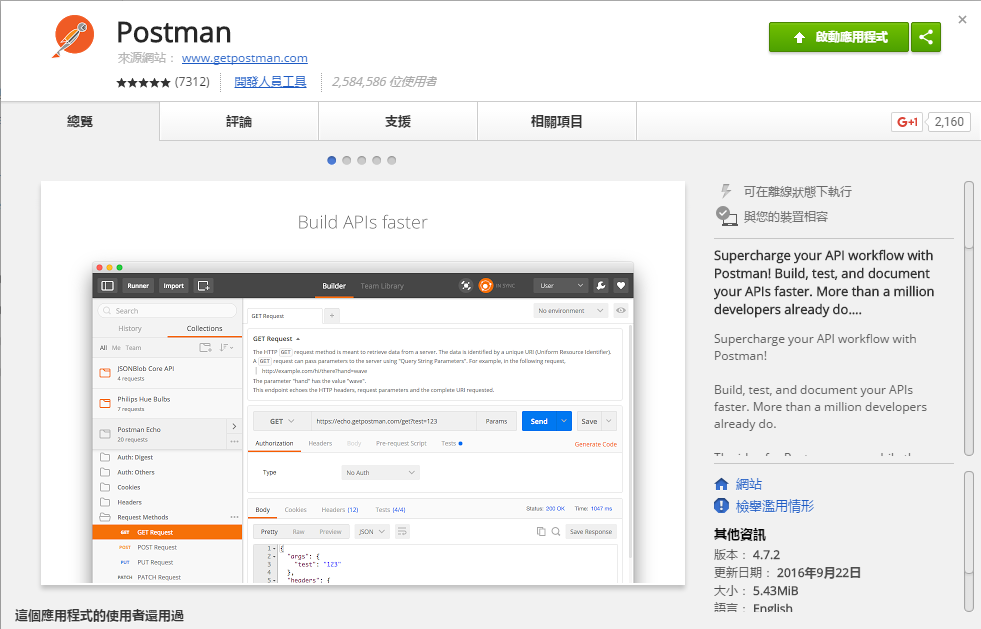
* + 1. Web Service 存取權限範圍
       1. 使用Client Credentials 方式驗證

為維護系統安全，若使用Client Credentials的方式驗證，傳送Client Id與Client Secret所取得的Token，僅能存取該Client Id所屬水資源物聯網 User本身及其所被分享的監測資訊。

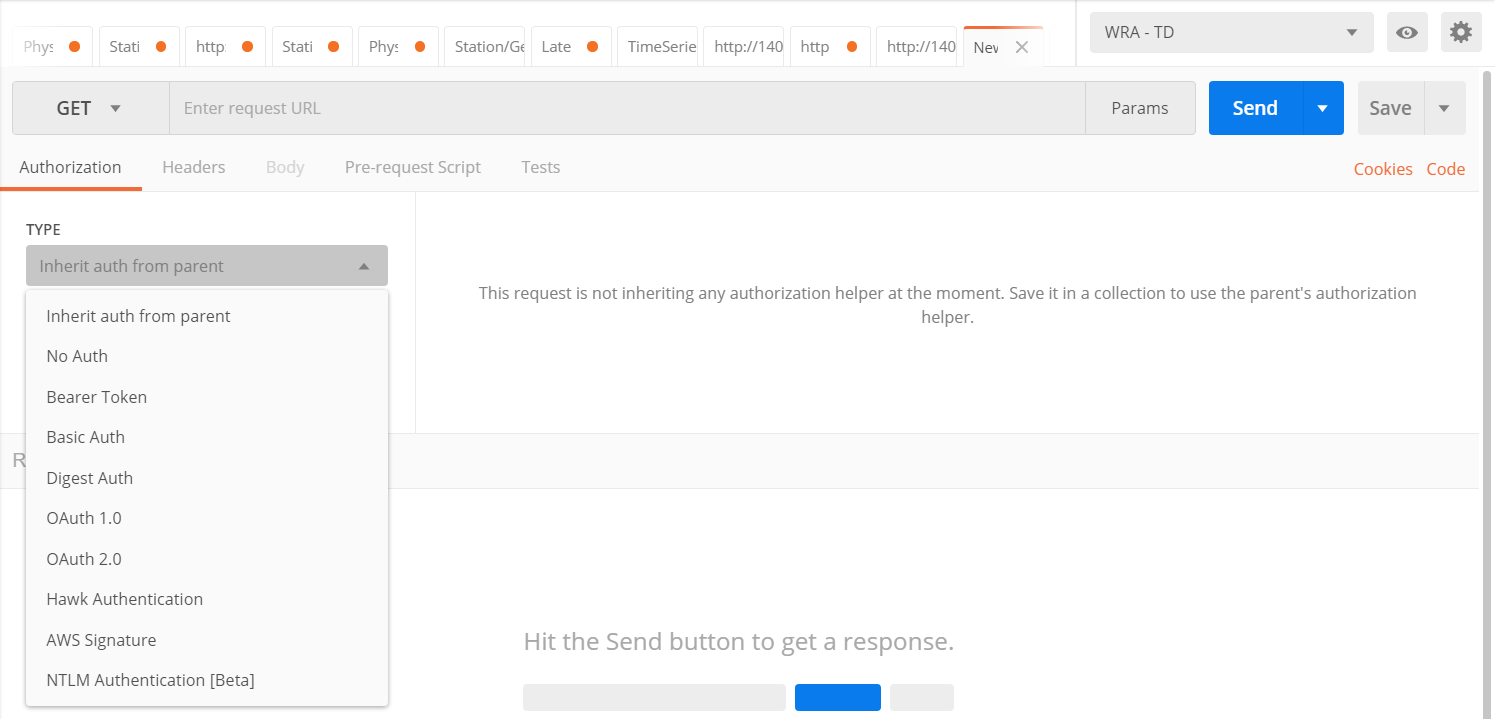
* + - 1. 使用Authorization Code方式驗證

使用Authorization Code驗證，當系統帶出水資源物聯網 User驗證帳密輸入畫面，可以輸入任何水資源物聯網 User帳密，所取得的Web Service讀取權限，則為該水資源物聯網 User的監測資訊。

* + 1. Client Credentials驗證範例
       1. 使用工具Token取得範例
          1. 安裝Google Chrome商店中的Postman。



* + - * 1. 設定認證協定為OAuth 2.0

1. 
   * + - 1. 點選Get New Access Token按鈕

並依序填入以下欄位資料

Auth URL: [空白]

Access Token URL: https://{root}/v3/oauth2/Token

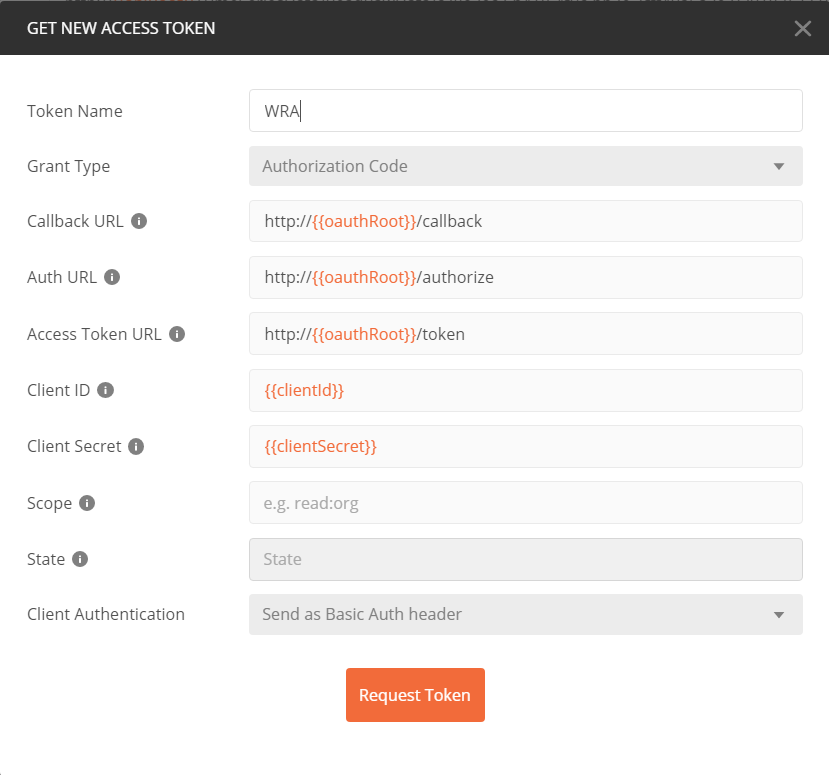
Client ID: [使用水資源物聯網網站取得的Client Id]

Client Secret: [使用水資源物聯網網站取得的Client Secret]

Scope: [空白]

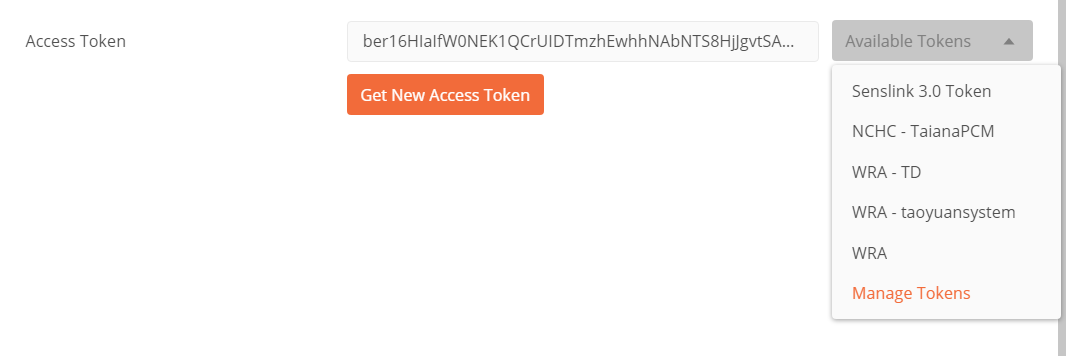
Grant Type: Client Credentials

按下Request Token，若認證成功，會成功取得Token。

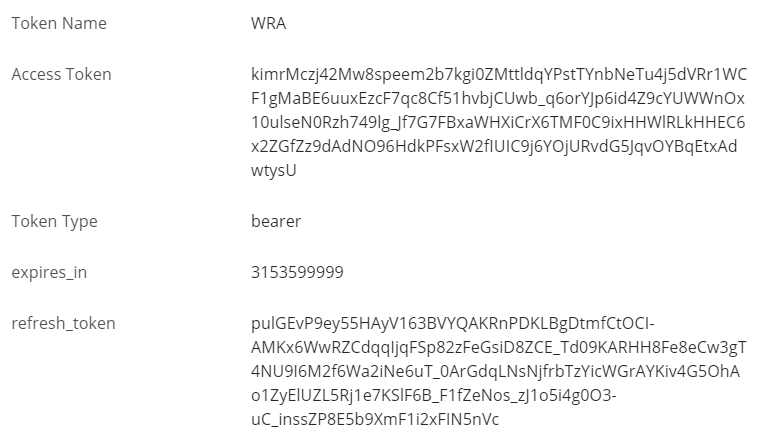


* + - * 1. 成功取得認證

於Existing Token視窗，會出現水資源物聯網 Token列表



點選後，右方視窗會出現所取得的Token，點選Add token to，選擇Header，作為取得水資源物聯網 Rest API相關資源做準備。



* + - 1. 使用Token，取得水資源物聯網 Rest API 資源

設定欲取得水資源物聯網 Rest API的資源路徑，例如

https://{root}/v3/api/station/get/all/

點選Send，客戶端軟體(以本例來說是Postman)實際會送出以下資訊至Rest API伺服器

GET /v3/api/station/get/all HTTP/1.1

Host: iapi.wra.gov.tw

Authorization: Bearer ◼◼◼◼◼◼◼◼◼◼◼◼◼P7Y6quT96SP4RtyeiJKdrwC7wLikKCUxjias3uOqW\_iUVxb0dEfrxQspoWUx003ipJvyS1a5vCdeCW4kb-5MmYJoRBBt0pBeOaWewv4IbmbiJ5zSXFZ3MhYebadkQU7dBdLckoarm20c5gErJdd4bhJQ7DDjyEeXeq1R7TG7OxDvMv0BKqtQEBL4juv4e5WmnIjR9sT6qMilZrrzA3eP

Cache-Control: no-cache

Rest API伺服器會回傳該資源應有的回應。

* 1. Web Service API 功能
     1. 命名規則

所有API 功能命名，結構為[動詞][結果/目標][給定條件]\_[補充說明]

[動詞] 對水資源物聯網系統的動作，如寫入或取得

[結果/目標] 該動作執行後期望獲得的結果，如取得監測站基本資料 (StationInfo)

[給定條件] 欲達成上述動作需給水資源物聯網系統的條件資料，例如Station Id (StationId)

[補充說明] 針對上述動作的補述，例如，取得即時資料，但是僅有AI，因此加上AIOnly

* + 1. 輸入參數與回傳型態使用注意事項

回傳資料之資料型態，若非該程式語言所定義的型態，則需自訂資料結構接收回傳值，資料結構定義參考(以c#為範例)。

* + 1. 子物件取用說明

Web Service API中，若需取得子項結構，使用 { ChildName1 | ChildName2 | …}之語法，例如，若需取得所有監測站及其所包含的物理量及物理量分類，可以使用

http://{root}/v3/api/Station/Get/All/**{PhysicalQuantity | Category}**

參數{PhysicalQuantity | Category}中的大括號，是必要符號。

* + 1. REST API 功能及路徑總表

API 路徑為 https://{root}/v3/api {主功能路徑}/{子功能路徑}

詳細API回傳資料結構詳見 https://{root}v3/api

|  |  |  |
| --- | --- | --- |
| REST API 主/子功能路徑表 | 功能 | 章節 |
| 監測站群組 主功能路徑 /StationGroup | | |
| /Get/All | 取得所有監測站群組資訊 | 6.5.1 |
| /Get/{StationGroup Id} | 使用監測站群組Id，取得監測站群組資訊 | 6.5.3 |
| /Get/{StationGroup Id}/{IncludedChilds} | 使用監測站群組Id，取得監測站群組資訊及其子項 | 6.5.4 |
| /Get/Station/{Station Id}/{IncludedChilds} | 使用監測站Id，取得監測站群組資訊 | 6.5.5 |
| 監測站 主功能路徑 /Station | | |
| /Get/All | 取得所有監測站資訊 | 6.6.1 |
| /Get/All/{IncludedChilds} | 取得所有監測站資訊及其子項 | 6.6.2 |
| /Get/{Station Id} | 使用監測站Id，取得特定監測站資訊 | 6.6.3 |
| /Get/{Station Id}/{IncludedChilds} | 使用監測站Id，取得特定監測站資訊及其子項 | 6.6.4 |
| 物理量 主功能路徑 /PhysicalQuantity | | |
| /Get/All | 取得使用者所有監測站下所有物理量資訊 | 6.7.1 |
| /Get/All/{IncludedChilds} | 取得使用者所有監測站下所有物理量資訊及其子項 | 6.7.2 |
| /Get/{PhysicalQuantity Id} | 使用某個監測站Id取得該監測站下所有物理量資訊 | 6.7.3 |
| /Get/{PhysicalQuantity Id}/{IncludedChilds} | 取得某一監測站群組下所有監測站物理量資訊及其子項 | 6.7.4 |
| 讀取最新物理量 /LatestData | | |
| /Read/All | 讀取所有物理量最新監測資料 | 6.8.1 |
| /Read/PhysicalQuantity/{PhysicalQuantity Id} | 讀取某一物理量最新監測資料 | 6.8.2 |
| /Read/PhysicalQuantity/{PhysicalQuantity Id}/{timeZoneInMinute} | 讀取某一物理量最新監測資料，並將回傳資料調整為所指定的時區 | 6.8.3 |
| /Read/Station/{Station Id} | 讀取某一監測站最新監測資料 | 6.8.3 |
| /Read/Station/{Station Id}/{timeZoneInMinute} | 讀取某一監測站最新監測資料，並將回傳資料調整為所指定的時區 | 6.8.5 |
| /Read/StationGroup/{StationGroup Id} | 讀取某一監測群組之最新監測資料 | 6.8.6 |
| /Read/StationGroup/{StationGroup Id}/{timeZoneInMinute} | 讀取某一監測群組之最新監測資料，並將回傳資料調整為所指定的時區 | 6.8.7 |
| 讀寫歷史物理量 /TimeSeriesData | | |
| /ReadAggregatedData/{PhysicalQuantity Id}/{StartDT}/{EndDT}/{AgMethod}/{AgInterval}/{TimeZone} | 讀取某一物理量歷史資料 | 6.8.8 |
| /ReadAggregatedData | 同時讀取多組歷史資料 | 6.8.9 |
| /ReadRawData/{physicalQuantityId}/{sDtStr}/{eDtStr}/{ignoreNaN} | 讀取原始資料 | 6.8.10 |
| /Write/ | 寫入物理量資料，會經過所設定的公式轉換進行運算。 | 6.8.11 |
| /Write/FormulaTransferred | 寫入物理量資料，不會經過所設定的公式轉換進行運算。 | 6.8.12 |
| /Delete/{PhysicalQuantity Id}/{StartDT}/{EndDT} | 刪除某一物理量一段歷史資料 | 6.8.13 |
| /ContemporaryComparison/{PhysicalQuantityId}/{ComparisonDt}/{DataInterval}/{CycleInterval}/{ComparisonCycles}/{TimeZoneInMinutes} | 同期比較，以同期過去n個Cycle作為母體，計算同期數值之標準差與平均數。 | 6.8.14 |
| 繪圖 /Chart | | |
| /Generate/{chartProvider} | 產生可嵌入互動式繪圖結果的短網址 | 6.9.1 |
| 取得Alarm設定及發送紀錄 /Alarm | | |
| /Get/Settings/PhysicalQuantity/{PhysicalQuantity Id} | 取得某一物理量之警報設定值 |  |
| /Get/Settings/Station/{Station Id} |  |  |

* + 1. 監測站群組 (ETL API)
       1. /StationGroups/Get/All
          1. HTTP Verb: GET
          2. 取得此帳號下所有監測站群組資訊
       2. /StationGroups/Get/All/{IncludedChilds}
          1. HTTP Verb: GET
          2. 取得此帳號下所有監測站群組資訊及其子項
          3. 子項包括Station, PhysicalQuantity
       3. /StationGroups/Get/{StationGroup Id}
          1. HTTP Verb: GET
          2. 使用監測站群組Id，取得特定監測站群組資訊
       4. /StationGroups/Get/{StationGroup Id}/{includedChilds}
          1. HTTP Verb: GET
          2. 使用監測站群組Id，取得特定監測站群組資訊及子項
          3. 子項包括Station, PhysicalQuantity
    2. 監測站 (ETL API)
       1. Station/Get/All
          1. HTTP Verb: GET
          2. 取得此帳號下所有監測站資訊
       2. Station/Get/All/{includedChilds}
          1. HTTP Verb: GET
          2. 取得此帳號下所有監測站資訊及其子項
          3. 子項包括 PhysicalQuantity, Category, AlarmSetting,
       3. /Station/Get/{Station Id}
          1. HTTP Verb: GET
          2. 取得此帳號下某一監測站資訊
       4. /Station/Get/{Station Id}/{includedChilds}
          1. HTTP Verb: GET
          2. 取得此帳號下某一監測站資訊及其子項
          3. 子項包括 PhysicalQuantity, Category, AlarmSetting
    3. 物理量 (ETL API)
       1. /PhysicalQuantity/Get/All
          1. HTTP Verb: GET
          2. 取得此帳號下所有物理量
       2. /PhysicalQuantity/Get/All/{includedChilds}
          1. HTTP Verb: GET
          2. 取得此帳號下所有物理量及其子項
          3. 子項包括Category, AlarmSetting
       3. /PhysicalQuantity/Get/{PhysicalQuantityId}
          1. HTTP Verb: GET
          2. 使用物理量Id取得某一物理量
       4. /PhysicalQuantity/Get/{PhysicalQuantityId}/{includedChilds}
          1. HTTP Verb: GET
          2. 使用物理量Id取得某一物理量及其子項
          3. 子項包括Category, AlarmSetting
    4. 物理量資料 (ETL API)
       1. /LatestData/Read/All
          1. HTTP Verb: GET
          2. 讀取所有監測物理量最新數值
       2. /LatestData/Read/PhysicalQuantity/{PhysicalQuantity Id}
          1. HTTP Verb: GET
          2. 使用某一PhysicalQuantity Id，讀取該物理量數值
       3. /LatestData/Read/PhysicalQuantity/{PhysicalQuantity Id}/{timeZoneInMinute}
          1. HTTP Verb: GET
          2. 指定PhysicalQuantity Id，讀取該物理量數值，並且將時間標記調整為所指定時區
       4. /LatestData/Read/Station/{Station Id}
          1. HTTP Verb: GET
          2. 使用某一監測站Id，讀取該監測站所有物理量數值
       5. /LatestData/Read/Station/{Station Id}/{timeZoneInMinute}
          1. HTTP Verb: GET
          2. 使用某一監測站Id，讀取該監測站所有物理量數值，並將回傳資料調整為所指定的時區
       6. /LatestData/Read/StationGroup/{StationGroup Id}
          1. HTTP Verb GET
          2. 使用某一監測站群組Id，讀取該監測站群組下，所有監測站之所有物理量數值
       7. /LatestData/Read/StationGroup/{StationGroup Id}/{timeZoneInMinute}
          1. HTTP Verb GET
          2. 使用某一監測站群組Id，讀取該監測站群組下，所有監測站之所有物理量數值，並將回傳資料調整為所指定的時區
       8. /TimeSeriesData/ReadAggregatedData/{PhysicalQuantityId}/{sDtStr}/{eDtStr }/{AgMethod}/{AgInterval}/{TimeZone}
          1. HTTP Verb GET
          2. 讀取某一物理量之彙總運算完成之歷史時間序列資料，並以使用者所設定的顯示時區進行資料調整。
          3. 路徑參數

PhysicalQuantity: 物理量Id，為GUID String

sDtStr: 起始時間，格式為 {yyyy}-{MM}-{dd}T{HH}.{mm}.{ss}

eDtStr: 結束時間，格式為 {yyyy}-{MM}-{dd}T{HH}.{mm}.{ss}

AgMethod: 彙總運算方法，為整數，參考資料結構定義。

AgInterval: 彙總運算區間，整數，單位為秒

注意: 由於URL路徑不支援符號內，不支援使用’/’及’:’，因此使用’-’及’.’取代

* + - 1. /TimeSeriesData/ReadAggregateData
         1. HTTP Verb POST
         2. 讀取多組物理量之彙總運算完成之歷史時間序列資料，並以使用者所設定的顯示時區進行資料調整。
         3. POST Body

{

“physicalQuantityIds”: [ ] //JSON ID Array

“startDt”: //string in ISO8601

“endDt”: //string in ISO8601

“agInterval”: //integer in seconds

“timeZoneInMinute” //integer in minute

}

* + - 1. /ReadRawData/{physicalQuantityId}/{sDtStr}/{eDtStr}/{ignoreNaN}/{timeZoneInMinute}
         1. HTTP Verb Get
         2. 路徑參數

PhysicalQuantity: 物理量Id，為GUID String

sDtStr: 起始時間，格式為 {yyyy}-{MM}-{dd}T{HH}.{mm}.{ss}

eDtStr: 結束時間，格式為 {yyyy}-{MM}-{dd}T{HH}.{mm}.{ss}

ignoreNaN: 是否不要回傳數值為NaN的資料點，若為True，則不回傳。

注意: 由於URL路徑不支援符號內，不支援使用’/’及’:’，因此使用’-’及’.’取代

* + - 1. /TimeSeriesData/Write
         1. HTTP Verb: POST  
            Header: Content-Type: Application/Json  
            寫入一至多數筆資料至某個物理量
         2. 傳入參數HTTP Post Body 之JSON Format範例

[

{

“Id” : “CA761232-ED42-11CE-BACD-00AA0057B223”,

TimeStamp” : “2016/5/10 12:30:01 +08:00”,

“Value” : 3.24597

},

{

“Id” : “CA761232-ED42-11CE-BACD-00AA0057B223”,

“TimeStamp” : “2016/5/10 12:31:01 +08:00”,

“Value” : 3.04597

}

]

* + - 1. /TimeSeriesData/Write/FormulaTransferred
         1. HTTP Verb: POST

Header: Content-Type: Application/Json  
寫入一至多數筆資料至某個物理量，但寫入時會略過系統上所設定的公式轉換，直接寫入資料庫。

* + - * 1. 傳入參數HTTP Post Body: 與6.8.11相同。
      1. /TimeSeriesData/Delete/{PhysicalQuantityId}/{StartDT}/{EndDT}
         1. HTTP Verb: DELETE
         2. 路徑參數

PhysicalQuantity Id: 物理量Id，為GUID String

StartDT: 起始時間，格式為 {yyyy}-{MM}-{dd}T{HH}.{mm}.{ss}

EndDT: 結束時間，格式為 {yyyy}-{MM}-{dd}T{HH}.{mm}.{ss}

* + - 1. /ContemporaryComparison/{PhysicalQuantityId}/{ComparisonDt}/{DataInterval}/{CycleInterval}/{ComparisonCycles}/{TimeZoneInMinutes}
         1. HTTP Verb: GET
         2. 路徑參數

PhysicalQuantityId: 物理量Id，為GUID String

ComparisonDt: 同期比較基準時間，格式為  
{yyyy}-{MM}-{dd}T{HH}.{mm}.{ss}

DataInterval: 用以計算同期比較之標準差、平均數的資料間距。

CycleInterval: 同期比較週期間距。

ComparisonCycles: 同期比較週期數

TimeZoneInMinutes: 以分鐘表示的時區

* + 1. 圖表 (可視化API)
       1. /Chart/Generate/{chartProvider}
          1. HTTP Verb: POST
          2. 路徑參數

chartProvider: 定義在Body中的Chart的繪製參數，是屬於哪一個Chart套件，目前系統支援兩種套件，包括ChartJs(<https://www.chartjs.org/>)，與Plotly(<https://plot.ly/javascript/>)，給定此參數即為ChartJS與Plotly。

* + - * 1. Http Body

Body的資料型態，可以為Json或是JavaScript，由Post時的MediaType給定，因此兩種套件繪圖支援這兩種繪圖模式。

使用Javascript

使用ChartJS或Ploty套件，Javascript繪製圖形指令，例如Plotly.newPlot(‘diagram’,data,layout)，繪圖輸入div，名稱必定為diagram。

例如

var data = [

{

x: ['2013-10-04 22:23:00', '2013-11-04 22:23:00', '2013-12-04 22:23:00'],

y: [1, 3, 6],

type: 'scatter'

}];

Plotly.newPlot('diagram', data);

使用Json

使用ChartJS套件時，直接將ChartJS相對應圖表的JSON格式填入Body即可，若為Plotly，則必須填入以下格式

{

“data” : [ ],

“layout”:

}

其中data和layout的格式，詳見Plotly說明。

* + - * 1. 回傳結果

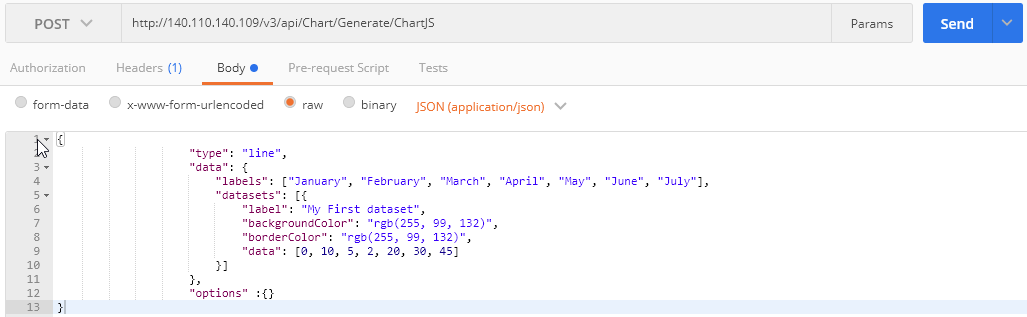
執行上述連結後，系統會回傳一連結，此連結可以嵌入任何網頁div，包括任何圖台之pop up視窗，形式如下

/Chart/Display/{UUID}

點選此連結系統會根據之前在Body所傳遞的內容進行繪圖，此連結若超過30日沒有任何人使用會自動失效。

* + - * 1. 範例

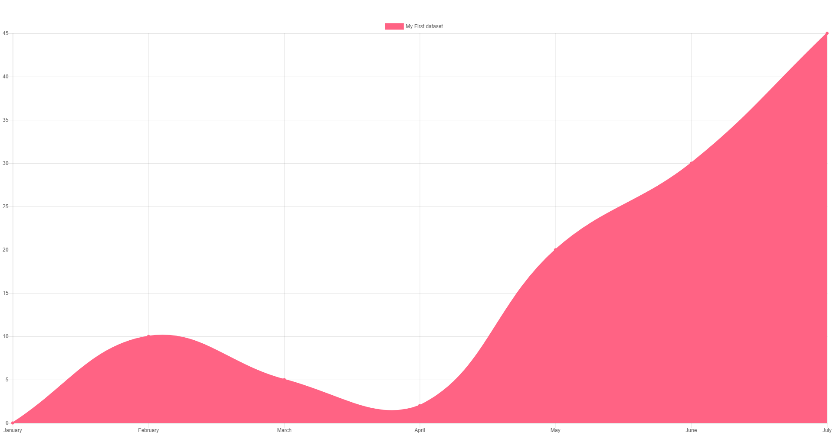
傳入Body



回傳連結

http://140.110.140.109/v3/api/display/{UUID}

繪圖範例結果



* + 1. 自定分散式運算 (雲端運算API)
       1. /Spark/Job/Status/{jobId}/{account}

使用jobId，查詢已經分配的job工作狀態

* + - * 1. HTTP Verb: Get
        2. 路徑參數

account

Spark使用者帳號

jobId

所發送的工作Id

* + - * 1. 回傳結果
      1. /Spark/Job/List/{account}

使用Spark使用者帳號，列出該帳號下目前所有工作狀態

* + - * 1. HTTP Verb: Get
        2. 路徑參數

account

Spark使用者帳號

* + - 1. /Spark/Job/Submit

分配工作至Spark分散式運算系統

* + - * 1. HTTP Verb: POST
        2. HTTP Body 格式

{

"Spark": {

"Account": "20180606",

"MainFile": "MyJob1.py",

"Driver": "10.57.234.251",

"Worker": [

"10.57.234.249",

"10.57.234.250",

"10.57.234.251"

],

Priority = "normal"

},

"S3": {

"InputPath": "MyJobs/MyJob1.py",

"SecretKey": "◼◼◼◼◼◼◼Lq33Bx2KfBWZZhyZkWYpwlgnMC3kWsy2",

"AccessKey": "◼◼◼◼◼◼◼0NMV060E59S3I"

}

}

* + - * 1. HTTP Body 參數說明

Spark.Account

Spark 帳號

Spark.MainFile

欲上傳/發送的分散式程式檔案名稱，此程式必須以Python(\*.py)或是Java(.jar)撰寫，並且事先上傳至S3儲存空間。  
若欲執行的工作是由多個Python檔案組成，則包裝為zip檔案，則MainFile代表zip檔案中的Python程式進入點。

Spark.Driver

Spark 分散式運算系統 Driver node的IP位置，此位址在申請此服務時取得。

Spark.Workers

Spark 分散式運算系統 Worker nodes的IP位置，此位址在申請此服務時取得，可以有多組，因此此參數為JSON Array。

S3.InputPath

位於S3檔案儲存空間，欲發送/上傳的程式路徑與完整名稱。

S3.SecretKey

存取S3檔案儲存空間所需之SecretKey。

S3.AccessKey

存取S3檔案儲存空間所需之AccessKey。

* + - * 1. 回傳參數

工作發送後，會回傳jobId。

* + - 1. /Spark/Job/Delete/{account}/{jobId}

刪除某個正在執行中的工作，若該工作正在排程，則會立即被移除，若已經在執行，則會等到該工作執行完畢。

* + - * 1. HTTP Verb: Delete
        2. 參數說明

account

Spark帳號

jobId

經由Submit方法回傳的Job Id

* + 1. 定型分散式運算 (大數據運算API)

本API提供9種定型化的大數據運算功能。

* + - 1. /Spark/Model/Train
         1. HTTP Verb: POST
         2. HTTP Body

{

"Model": {

"Name": {Name}, //Model Name

"Parameters": //Model training parameters

},

"S3": {

"TrainingData": "MyData/trainingData.csv",

"ModelOutput" : "MyModel/ModelResult.mdl",

"SecretKey": "◼◼◼◼◼◼◼Lqx2KfBWZZhyZkWYpwlgnMC3kWsy2",

"AccessKey": "◼◼◼◼◼◼◼0NMV060E59S3I"

}

}

* + - * 1. HTTP Body 參數說明

Model.Name : 欲執行的演算法名稱。

Model.Parameters: 欲執行的演算法所支援的訓練調整參數。

S3.TrainingData: 訓練模型所使用的資料。

S3.ModelOutput: 訓練結果輸出路徑與檔案名稱。

S3.SecretKey: 連接 S3檔案系統的SecretKey

S3.AccessKey: 連接S3檔案系統的AccessKey

* + - 1. /Spark/Model/Consume
         1. HTTP Verb: POST
         2. HTTP Body

{

"Model": {

"Name": {Name}, //Model Name

},

"S3": {

"Model" : "MyModel/ModelResult.mdl",

"Input": "",

"Result" : "MyResult/Result.txt",

"SecretKey": "◼◼◼◼◼◼◼Lqx2KfBWZZhyZkWYpwlgnMC3kWsy2",

"AccessKey": "◼◼◼◼◼◼◼0NMV060E59S3I"

}

* + - * 1. HTTP Body 參數說明

Model.Name : 欲執行的演算法名稱。

S3.TrainingData: 訓練模型所使用的資料。

S3.Model: 已經儲存的訓練結果路徑與檔案名稱。

S3.Result: 計算結果。

S3.SecretKey: 連接 S3檔案系統的SecretKey

S3.AccessKey: 連接S3檔案系統的AccessKey

* + 1. 點陣地理圖資

上傳運算完畢後之點陣地理圖資資料，以國際標準NetCDF儲存，並提供相關Web Service可運算及操作二維地理點陣圖資

* + - 1. RasterMap/File/List

列出檔案

* + - * 1. HTTP Verb: Get
        2. 路徑參數

無

* + - * 1. 回傳結果

[

{

"Id": "0b1bf460-a458-4ae9-b710-d939a9f2dbfd", //檔案ID

"FileName": "myFile.nc", //名稱

"Owner": "me", //擁有者

"SharedLevel": "Public", //檔案分享權限

"Download": //下載連結"http://{root}/RasterMap/File/Download/me/myFile.nc/",

"Delete": //刪除連結 "http://{root}/RasterMap/File/Delete/myFile.nc/",

"NetCDF ASCII": //以NetCDF ASCII格式回傳

"http://{root}/RasterMap/Get/me/myFile.nc/",

"WMS/Capabilities": //WMS Capability 資訊 "http://{root}/RasterMap/WMS/Get/Capabilites/myFile.nc/",

"WMS/ImageInfos": //取得圖檔資訊"http://{root}/RasterMap/Get/ImageInfos/myFile.nc/",

"Godiva3Viewer": //地理圖資套疊測試連結，使用Godiva圖台

"http://{root}/..."

}

]



上傳檔案

* + - * 1. HTTP Verb: Put
        2. 路徑參數

overWrite : 若上傳檔名相同，是否覆寫，True為覆寫，False不覆寫

sharedWithUsers : 此檔案欲分享給其他使用者名稱，使用逗點(,)分隔，若分享名稱包含保留字public，則表示所有人都可以讀取，若為private，則僅有自己可以讀取。

* + - 1. RasterMap/File/Download/{fileName}

下載屬於自己的檔案

* + - * 1. HTTP Verb: Get
      1. RasterMap/File/Download/{ownerName}/{fileName}

下載其他帳號分享給自己的檔案

* + - * 1. HTTP Verb: Get
      1. RasterMap/File/Delete/{fileName}

刪除檔案

* + - * 1. HTTP Verb: Get
      1. RasterMap/NetCDF/Get/{ownerName}/{fileName}

以ASCII格式回傳整個NetCDF檔案。

* + - * 1. HTTP Verb: Get
      1. RasterMap/NetCDF/GetGrid/{fileName}/{variableName}/{unixtimeInMins}/{leftUpCornerX}/{leftUpCornerY}/{rightDownCornerX}/{rightDownCornerY}

以ASCII格式回傳部分NetCDF檔案

* + - * 1. HTTP Verb: Get
      1. RasterMap/NetCDF/GetGrid

以ASCII格式回傳部分NetCDF檔案

* + - * 1. HTTP Verb: Post
      1. RasterMap/NetCDF/Execute

執行NetCDF二維運算命令，可執行之二維命令，詳見

<http://docs.opendap.org/index.php/Server_Side_Processing_Functions>

* + - * 1. HTTP Verb: POST
        2. Body

{

“fileName” : “myFile.nc” //欲執行運算的NetCDF檔案名稱

“query” : “” //欲執行運算的命令

}

* + - 1. WMS/Get/Capabilites/{fileName}

取得此NetCDF之WMS詮釋資料連結，可套疊至其他圖台。

* + - * 1. HTTP Verb: Get
      1. WMS/Get/ImageInfos/{fileName}

取得此NetCDF轉換之影像檔案資訊

* + - * 1. HTTP Verb: Get
  1. 資料結構定義
     1. 列舉型態
        1. AgMethod彙總運算方法

1: 平均

2: 加總

3: 積分

4: 最大值

5: 最小值

6: 計算數量

7: 第一筆

8: 最後一筆(時間最晚)

9: 區間最後一筆與上個區間最後一筆差值

10: 最小平方法

11: 百分位

12: 標準差

* 1. 使用 R 讀取 水資源物聯網 Web Service 範例
     1. 一般說明
        + 1. 以下{ }表示為須取代的變數
     2. 所需套件

curl, jsonlite, httr

* + 1. 取得Token

url <- "http://{root}/v3/oauth2/token"

body <- list(grant\_type = "client\_credentials", client\_id = "**{Id}**" , client\_secret = "**{Password}**")

req <- POST(url, body = body, encode = 'form')

jsonContext <- content(req, as = "text")

result = fromJSON(jsonContext)

tokenString = result$access\_token

* + 1. 讀取彙總歷史資料
       1. 通用格式

token <- paste("Bearer","**{tokenString}**")

url <-  
"http://{root}/v3/api/TimeSeriesData/ReadAggregatedData/**{physical Quantity Id}** /**{start date time}**/**{end date time}**/**{aggregate method}**/**{aggregate Interval}**"

req <- GET(url, add\_headers(Authorization = token))

jsonContext <- content(req, as = "text")

result = fromJSON(jsonContext)

* + - 1. 範例

token <- paste("Bearer",  
"1c\_N4QNfihyRtb5rigL5zTDAiPMCymflMfRBr1l5LfUvxIP0cGpyj3oZISV8ByrwdYaQElHydf4Meas0koO9tctVnDDYhfbG\_KF-yeP2EVuExkCrSMee4d32vOqrzJOBqCf91N0mrsNrCNSfQkYWatH8-Y0u7sTBqQClS0DMUSRYrN65gxJcTzIV66Are-essb3FXnEoiI4RpGy1YpSoXbIS9ZOq6YStnAM\_OBm-jwAlZ9WD")

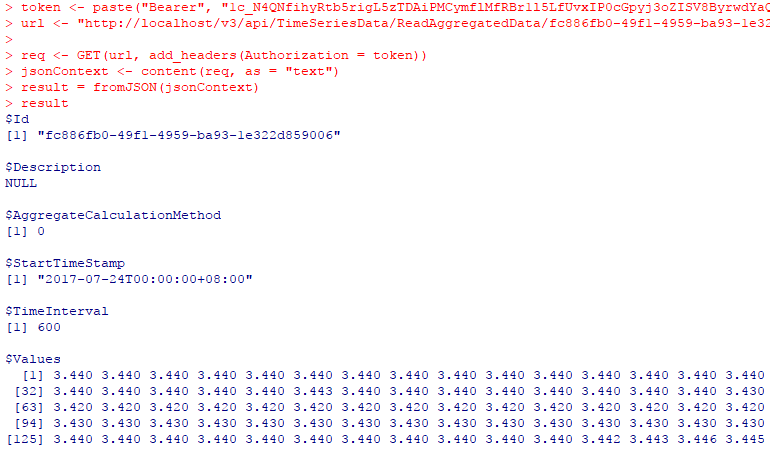
url <-

"https://{root}/v3/api/TimeSeriesData/ReadAggregatedData/fc886fb0-49f1-4959-ba93-1e322d859006/2017-07-24T00.00.00/2017-07-25T00.00.00/1/600"

req <- GET(url, add\_headers(Authorization = token))

jsonContext <- content(req, as = "text")

result = fromJSON(jsonContext)



* 1. 使用Python讀取 水資源物聯網 Web Service 範例
     1. 所需套件

json, requests

* + 1. 取得Token

import json

import requests

payload = { 'grant\_type':'client\_credentials', 'client\_id':**’{Id}’**, 'client\_secret':**’{password}’** }

response = requests.post("http://**{root}**/v3/oauth2/token", data=payload).josn()

token = response['access\_token']

* + 1. 讀取所有監測站資訊

以下範例需接續以上取得Token程式

headers = {"Accept": "application/json", 'Authorization':'Bearer %s'%token}

response = requests.get("http://**{root}**/v3/api/Station/Get/All", headers = headers).json()