



巨匠線上真人

# Java Web OCE JWCD元件系統 開發認證

[www.pcschoolonline.com.tw](http://www.pcschoolonline.com.tw)



巨匠線上真人

Java Web OCE JWCD元件系統開發認證

## 第十二堂

# JavaBean與Entity設計與應用

# 本堂教學重點

- ◆ JavaBean用途
- ◆ 使用Action Tag使用JavaBean於網頁中
- ◆ 實現序列化機制，JavaBean序列成JSON整合雲端服務
- ◆ 實務整合CHT IoT智慧聯網-傳遞一個溫濕度JSON資訊除儲存

# JavaBean用途

- ◆ JavaBean為特殊類別規劃，一般Java Root類別Object為父類別。
- ◆ 具有空參數建構子。
- ◆ 可以進行序列化進行傳輸或者儲存。
- ◆ 具有封裝性的Attribute，可以進行資料儲存。
- ◆ 同時透過setter與getter存取封裝欄位。
- ◆ 可為元件(Component)，因此Java稱呼元件為Bean。

## Customer.java

```
package com.gjun.domain;
//JavaBean
public class Customer implements java.io.Serializable{
    //attribute
    private short customerId;
    private String firstName;
    private String lastName;
    private String email;

    //空參數建構子
    public Customer() {
    }

    //setter and getter
    public short getCustomerId() {
        return customerId;
    }
    public void setCustomerId(short customerId) {
        this.customerId = customerId;
    }
    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    public String getLastName() {
        return lastName;
    }
}
```

# JavaBean用在哪裡？

customersform.html

- ◆ 可以稱呼為 Java元件。
- ◆ 可以是一個Entity Class應對的資料記錄架構。
- ◆ 個體物件可以是代表是一筆記錄。
- ◆ 可以是一個表單網頁的欄位對應的實體。

http://localhost:8080/mod12/customersform.html

客戶資料維護

客戶編號

Frist Name

Last Name

EMAIL

傳送

```
<fieldset>
  <legend>客戶資料維護</legend>
  <form method="post" action="">
    <div>客戶編號</div>
    <input type="text" name="customerId"/>
    <div>Frist Name</div>
    <input type="text" name="firstName"/>
    <div>Last Name</div>
    <input type="text" name="lastName"/>
    <div>EMAIL</div>
    <input type="text" name="email"/>
    <br/>
    <input type="submit" value="傳送"/>
  </form>
</fieldset>
```

```
package com.gjun.domain;
//JavaBean
public class Customer implements java.io.Serializable{
  //attribute
  private short customerId;
  private String firstName;
  private String lastName;
  private String email;

  //空參數建構子
  public Customer() {
  }

  //setter and getter
  public short getCustomerId() {
    return customerId;
  }
  public void setCustomerId(short customerId) {
    this.customerId = customerId;
  }
  public String getFirstName() {
    return firstName;
  }
  public void setFirstName(String firstName) {
    this.firstName = firstName;
  }
  public String getLastName() {
    return lastName;
  }
}
```

# 使用Action Tag使用JavaBean於網頁中

- ◆ 使用Action Element進行表單欄位自動封裝成一個JavaBean物件。
- ◆ 透過JavaBean Property(setter and getter)與表單欄位name attribute對應名稱設定，可以自動封裝欄位內容，產生一個實體物件。
- ◆ 方便進行資料維護與呈現作業。

customersadd.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>客戶資料維護</title>
</head>
<body>
<jsp:useBean id="customer" class="com.gjun.domain.Customer"></jsp:useBean>
<jsp:setProperty property="*" name="customer"/>
<div>客戶編號:${customer.customerId}</div>
<div>First Name:${customer.firstName}</div>
<div>Last Name:${customer.lastName}</div>
<div>EMAIL:${customer.email}</div>
</body>
</html>
```

# Action Element

## Custom DAO與DataSource整合

- ◆ 自訂DAO類別透過Action Element產生一個Instance物件。
- ◆ 採用action elements與EL進行資料新增作業。

客戶資料維護

客戶編號

900

First Name

Eric

Last Name

Chen

EMAIL

chen@cht.com.tw

傳送



http://localhost:8080/mod12/customersadd.jsp

true

客戶編號:900

First Name:Eric

Last Name:Chen

EMAIL:chen@cht.com.tw

customersadd.jsp

```
<body>
<%
//建構DataSource物件
BasicDataSource dataSource=new BasicDataSource();
dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");
dataSource.setUrl("jdbc:mysql://localhost:3306/sakila?useSSL=false"
+ "&serverTimezone=UTC&characterEncoding=utf-8");
dataSource.setUsername("root");
dataSource.setPassword("1111");
pageContext.setAttribute("datasource",dataSource);
%>
<jsp:useBean id="dao" class="com.gjun.domain.CustomerDao"></jsp:useBean>
<jsp:setProperty name="dao" property="dataSource" value="${datasource}"/>
<jsp:useBean id="customer" class="com.gjun.domain.Customer"></jsp:useBean>
<jsp:setProperty property="*" name="customer"/>

${dao.insert(customer)}

<div>客戶編號:${customer.customerId}</div>
<div>First Name:${customer.firstName}</div>
<div>Last Name:${customer.lastName}</div>
<div>EMAIL:${customer.email}</div>
</body>
```

# 何謂JSON

- ◆ SON ( JavaScript Object Notation , JavaScript物件表示法 , 讀作/'dʒeɪsən/ ) 是一種由道格拉斯·克羅克福特構想和設計、輕量級的資料交換語言，該語言以易於讓人閱讀的文字為基礎，用來傳輸由屬性值或者序列性的值組成的資料物件。儘管JSON是JavaScript的一個子集，但JSON是獨立於語言的文字格式，並且採用了類似於C語言家族的一些習慣。
- ◆ JSON 資料格式與語言無關。即便它源自JavaScript，但目前很多程式語言都支援 JSON 格式資料的生成和解析。JSON 的官方 MIME 類型是 application/json，副檔名是 .json。

```
{
  "events": [
    {
      "type": "message",
      "replyToken": "cc54eba0b1584264b7ec03b40cb27851",
      "source": {
        "userId": "U2f68594c974f342461e8363cc9cb8dcd",
        "type": "user"
      },
      "timestamp": 1582346027803,
      "mode": "active",
      "message": {
        "type": "text",
        "id": "11472842393379",
        "text": "您好"
      }
    }
  ],
  "destination": "U9398f923317ac66c18201b1954063b8b"
}
```

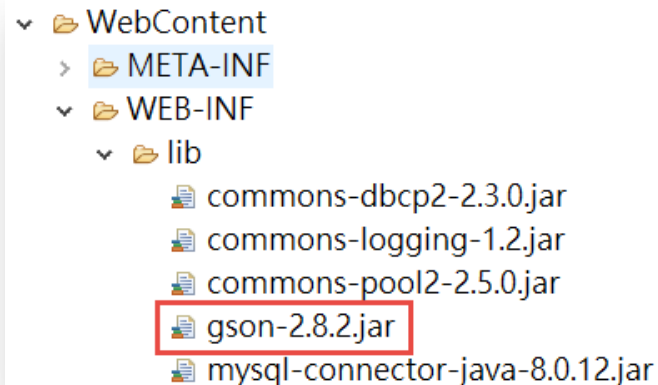
```
object {2}
  events [1]
    {6}
      type : message
      replyToken : cc54eba0b1584264b7ec03b40cb27851
      source {2}
        userId : U2f68594c974f342461e8363cc9cb8dcd
        type : user
      timestamp : 1582346027803
      mode : active
      message {3}
        type : text
        id : 11472842393379
        text : 您好
      destination : U9398f923317ac66c18201b1954063b8b
```



# 實現序列化機制

## JavaBean序列成JSON整合雲端服務

- ◆ JavaBean Implements `java.io.Serializable`。
- ◆ 借助Third Party API-Gson進行序列化與反序列化。
- ◆ 掛入Gson API到網站專案中。



# Servlet設計-查詢結果序列化為JSON



```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
```

```
    //擷取參數內容(查詢鍵值)
```

```
    String customerId=request.getParameter("cid");
```

```
    //建構DataSource
```

```
    BasicDataSource dataSource=new BasicDataSource();
```

```
    dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");
```

```
    dataSource.setUrl("jdbc:mysql://localhost:3306/sakila?useSSL=false"
        + "&serverTimezone=UTC&characterEncoding=utf-8");
```

```
    dataSource.setUsername("root");
```

```
    dataSource.setPassword("1111");
```

```
    //建構自訂的DAO物件
```

```
    IDao<Customer> dao=new CustomerDao();
```

```
    //入住DataSource
```

```
    dao.setDataSource(dataSource);
```

```
    //建構Customer物件
```

```
    Customer customer;
```

```
    //建構Customer物件
```

```
    Customer customer;
```

```
    try {
```

```
        customer = dao.selectForObject("select customer_id,first_name,last_name,email "
            + "0from customer where customer_id=?",
            , customerId);
```

```
    //建構GSON物件
```

```
    Gson gson=new Gson();
```

```
    String data=gson.toJson(customer); //序列化物件為JSON
```

```
    //設定回應Content-Type與內容
```

```
    response.setCharacterEncoding("UTF-8");
```

```
    response.setContentType("application/json");
```

```
    PrintWriter out=response.getWriter();
```

```
    out.println(data);
```

```
    } catch (SQLException e) {
```

```
        // TODO Auto-generated catch block
```

```
        e.printStackTrace();
```

```
    }
```

# 使用Postman測試

GET http://localhost:8080/mod12/CustomertoJsonServlet?cid=1

Params • Authorization Headers (7) Body Pre-request Script Tests

Query Params

KEY	VALUE
cid	1
Key	Value

Body Cookies Headers (3) Test Results

Pretty Raw Preview Visualize BETA JSON

```
1 {
2   "customerId": 1,
3   "firstName": "MARY",
4   "lastName": "SMITH",
5   "email": "MARY.SMITH@sakilacustomer.org"
6 }
```

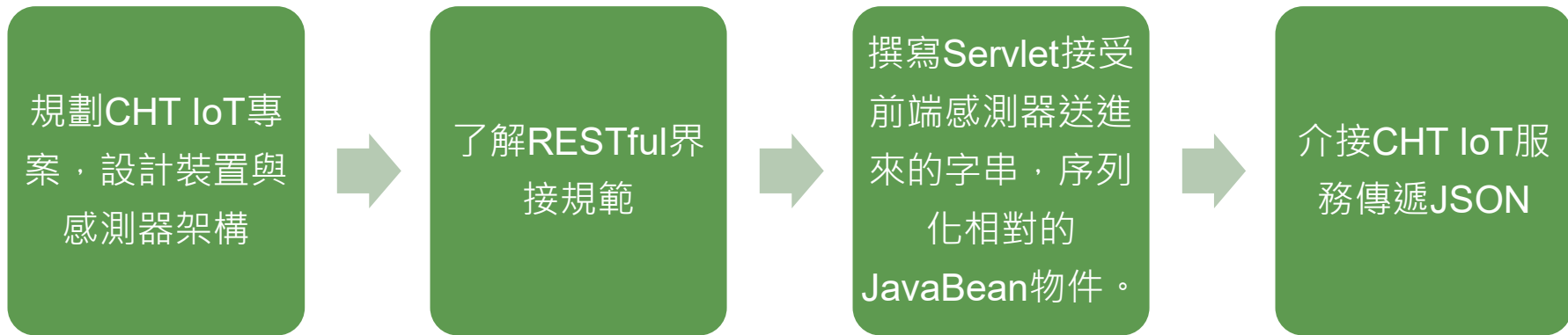
Body Cookies Headers (3) Test Results

KEY	VALUE
Content-Type ⓘ	application/json; charset=UTF-8
Content-Length ⓘ	96
Date ⓘ	Tue, 17 Dec 2019 00:32:52 GMT

# 實務整合CHT IoT智慧聯網

## -傳遞一個溫濕度JSON資訊除儲存

- ◆ 假設內部網站撰寫一個Servlet接受前端感測器送上來的溫濕度資訊。
- ◆ 接下將溫溼度資訊序列化JSON訊息，介接CHT IoT智慧平台，傳送上去JSON，並且可透過WebSocket推播到監控台系統去。
- ◆ 序列化與反序列化作業，借助JavaBean規範對應完成。



# 規劃CHT IoT專案，設計裝置與感測器架構

- ◆ 使用任何一支手機申請帳號。
- ◆ 建立專案，填寫專案資料，並且設定專案的權限(Key)。
- ◆ 入口
  - ◆ <https://iot.cht.com.tw/iot/login>

智慧聯網大平台  
SMART PLATFORM

歡迎登入

中華電信會員中心登入

帐号

密码

☐ 记住我的帐号

[忘记密码](#) [注册](#)

登入 [寬頻上網HN登入](#)

# 新增專案 設定權限



專案管理

專案名稱：  
JavaBean物聯網平台

專案描述：  
JavaBean物聯網平台

應用領域：  
其他

關閉 儲存



金鑰設定

專案金鑰	權限	備註
PKZA219SWM7H9KFYMH	read_write	System default configuration
PK7SXHGSG53WME1RXX	read_write	

新增權限 唯讀 (Read or) 備註 + 關閉

# 設定邏輯性的設備與實體對應的感測器

JavaBean物聯網平台 專案

搜尋設備名稱或描述...

設備編號	設備名稱	設備描述	設備類型	功能
尚未建立設備!				

顯示第 0 至 0 項結果

設備管理

基本資料 擴充屬性資訊

新增模式 ☒ 使用者設定 ☐ 從領域模板匯入

設備名稱

設備描述

設備類型 ☒ 通用設備 ☐ Modbus工業設備 ☐ UDP

經度  緯度

URI

IoT 智慧聯網大平台 SMART PLATFORM

開發者中心 專案管理 應用服務 聯絡我們 繁體中文

設備編號	設備名稱	設備描述	設備類型	功能
20627236882	教室溫溼度	教室溫溼度	general	<input type="button" value="編輯"/> <input type="button" value="複製"/> <input type="button" value="刪除"/>

顯示第 1 至 1 項結果

「教室溫溼度」設備資訊 (編號:20627236882)

感測器 設備內容 事件驅動 憑證申請 存取統計 連線管理 主動監控

共有 0 個感測器

感測器管理

基本資料 其他資料

識別編號(ID)  識別名稱具有唯一性

識別編號只允許輸入英文或數字或底線符號

顯示名稱

描述

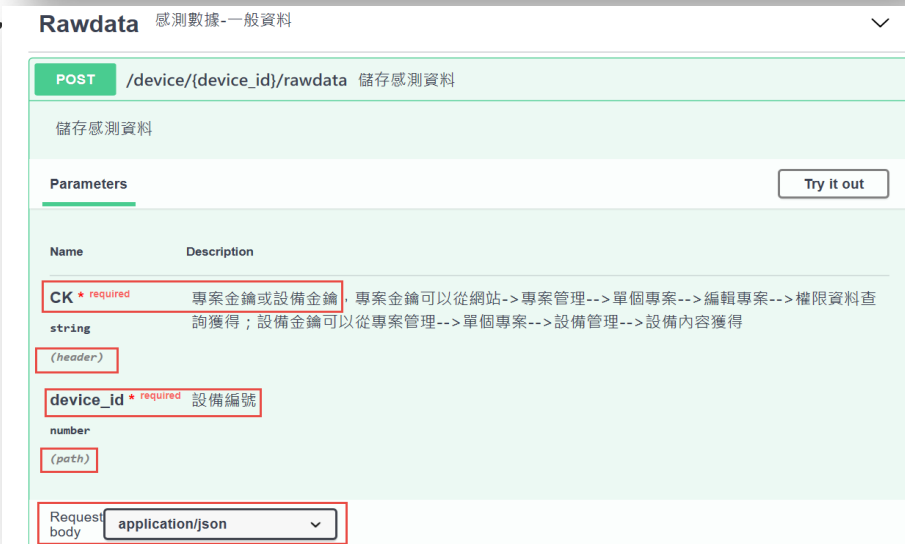
傳遞JSON文件格式儲存

類型 ☐ 數值 ☒ 文字 ☐ 開關 ☐ 圖像

# 了解RESTful界接規範

## ◆ 採用RESTful介接方式，必要要件：

- ◆ Hosted主機位址
- ◆ RESTful服務端點
- ◆ HTTP Request Method(傳送方法)，新增資料採用POST
- ◆ 請求時HTTP Header設定
  - API-KEY(金鑰)
  - Content-Type
- ◆ 傳送的Rawdata Http Body JSON文件格式





# CHT IoT 傳送感測資料JSON格式 對應JavaBean設計

SensorData.java

```
[
{
  "id": "temperature",
  "time": "2017-08-04T10:43:58Z",
  "lat": 24.95,
  "lon": 121.16,
  "save": true,
  "value": ["26.8"]
},
{
  "id": "humidity",
  "time": "2017-08-04T10:43:58Z",
  "lat": 24.95,
  "lon": 121.16,
  "save": true,
  "value": ["33"]
},
{
  "id": "pm",
  "time": "2017-08-04T10:43:58Z",
  "lat": 24.95
```

JSONArray

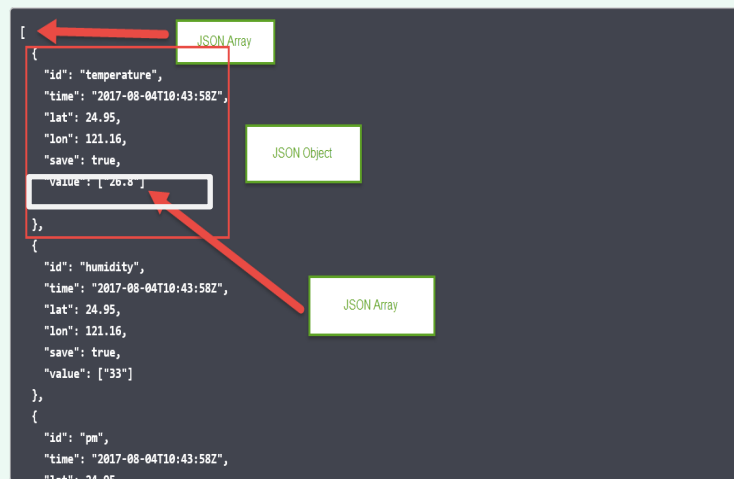
JSON Object

JSONArray

```
public class SensorData implements java.io.Serializable{
    //attribute
    private String id; //device id
    private String time; //ISO 8601 format
    private double lat; //緯度
    private double lon; //經度
    private boolean save; //儲存狀態
    private List<String> value; //多資訊字串(集合收集)
    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getTime() {
        return time;
    }
    public void setTime(String time) {
        this.time = time;
    }
    public double getLat() {
        return lat;
    }
}
```

# 自訂封裝溫溼度JavaBean

- ◆ 用來產生JSON String，封裝溫溼度資訊，且透過符合CHT IoT JSON文件格式的value進行集合收集。



- ◆ Value:[‘json string’]

```
package com.gjun.domain;
//溫溼度資料
public class DHTEntity implements java.io.Serializable {
    private double temper;
    private double humi;
    public double getTemper() {
        return temper;
    }
    public void setTemper(double temper) {
        this.temper = temper;
    }
    public double getHumi() {
        return humi;
    }
    public void setHumi(double humi) {
        this.humi = humi;
    }
}
```

DHTEntity.java

# 介接CHT IoT RESTful規範文件

要件	說明
Hosted入口主機位址	https://iot.cht.com.tw/iot/v1
新增感測資料End Point	/device/{device_id}/rawdata
Request Method	POST
Header	CK:XXXXXXXXXXXXXXXXXXXX(專案可讀寫資料key)
Header	Content-Type:application/json
Body(Raw Data)	JSON文件格式(請參考上一頁說明)

# 撰寫Servlet接受前端感測器送進來的字串 序列化相對的JavaBean物件。

- ◆ 採用QueryString傳遞前端溫濕度狀態進入Servlet。
- ◆ 建構對應CHT IoT JSON文件格式的泛型集合物件。

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    //擷取QueryString架構傳遞進來的溫濕度資料
    String temper=request.getParameter("temper");
    String humi=request.getParameter("humi");

    //建構DHT物件
    DHTEntity dht=new DHTEntity();
    dht.setTemper(Double.parseDouble(temper));
    dht.setHumi(Double.parseDouble(humi));

    //序列化JSON String
    Gson gson=new Gson();
    String dhtString=gson.toJson(dht);

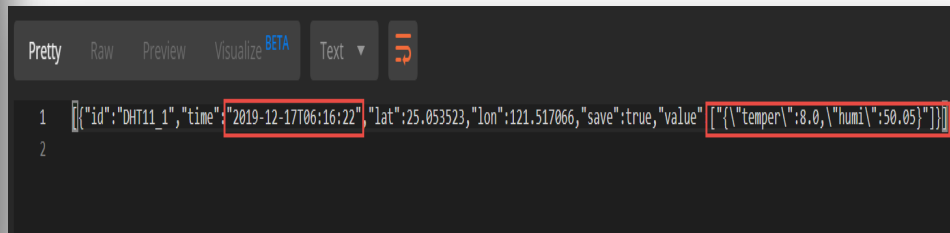
    //建構SensorData物件
    SensorData data=new SensorData();
    data.setId("DHT11_1");
    data.setLan(121.517066);
    data.setLat(25.053523);
    data.setSave(true);

    //系統時間轉換成ISO8601
    Calendar calendar = Calendar.getInstance();
    Date date = calendar.getTime();
    // 轉換ISO8601
    SimpleDateFormat sdf;
    sdf = new SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss");
    sdf.setTimeZone(TimeZone.getTimeZone("GMT"));
    String text = sdf.format(date);
    data.setTime(text);
}
```

# PostMan測試上傳CHT IoT的JSON文件內容

```
//建構集合物件
List<String> value=new ArrayList<>();
value.add(dhtString);
data.setValue(value);
//建構集合物件
List<SensorData> json=new ArrayList<>();
json.add(data);
//序列化JSON
String dataString=gson.toJson(json);
```

GET http://localhost:8080/mod12/DHTServlet?temper=8&humi=50.05



```
1 [{"id":"DHT11_1","time":"2019-12-17T06:16:22","lat":25.053523,"lon":121.517066,"save":true,"value":[{"temper":8.0,"humi":50.05}]}]
2
```

```
[{"id":"DHT11_1","time":"2019-12-17T06:16:22","lat":25.053523,"lon":121.517066,"save":true,"value":[{"temper":8.0,"humi":50.05}]}]
```

# 採用URL與URLConnection上傳資料

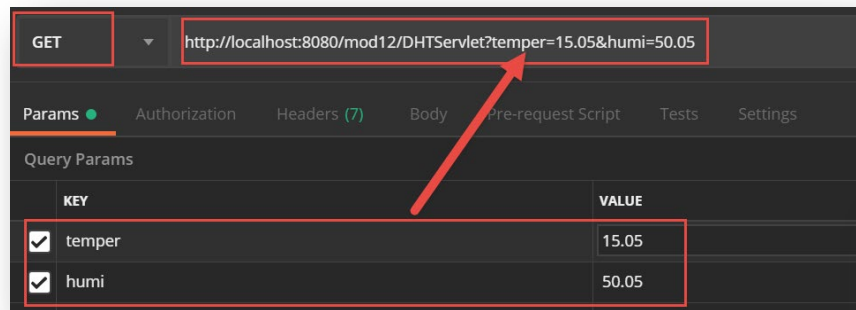
- ◆ 建構URL物件指定Hosted與端點
- ◆ 其中有一個path設定為Device ID
- ◆ 設定傳送方式與加入Header
  - ◆ 使用URLConnection  
setRequestProperty()設定。
- ◆ 設定Output與Input
- ◆ 透過連接物件參照出  
DataOutputStream，進行JSON文件  
內容寫出去(上傳)
- ◆ 取回InputStream讀取回應的結果內  
容。

```
//Http Client提出請求
URL url=new URL("https://iot.cht.com.tw/iot/v1/device/20627236882/rawdata");
URLConnection connection=(URLConnection)url.openConnection();
//設定傳送方式
connection.setRequestMethod("POST");
//設定Header
connection.setRequestProperty("Content-Type","application/json");
connection.setRequestProperty("CK","PK7SXHG53WME1RXX");
connection.setDoInput(true);
connection.setDoOutput(true);

//寫出資料
DataOutputStream outputStream = new DataOutputStream(connection.getOutputStream());
outputStream.write(dataString.toString().getBytes("UTF-8"));
outputStream.flush();
outputStream.close();

InputStreamReader reader = new InputStreamReader(connection.getInputStream());
BufferedReader br = new BufferedReader(reader);
String line = null;
StringBuilder builder=new StringBuilder();
while( (line = br.readLine()) != null ) {
    builder.append(line+"\n");
}
outputStream.close();
response.getWriter().println(builder.toString());
```

# PostMan測試-模擬溫濕度感測資料上傳



◆ 檢視CHT lot雲端接受的資料

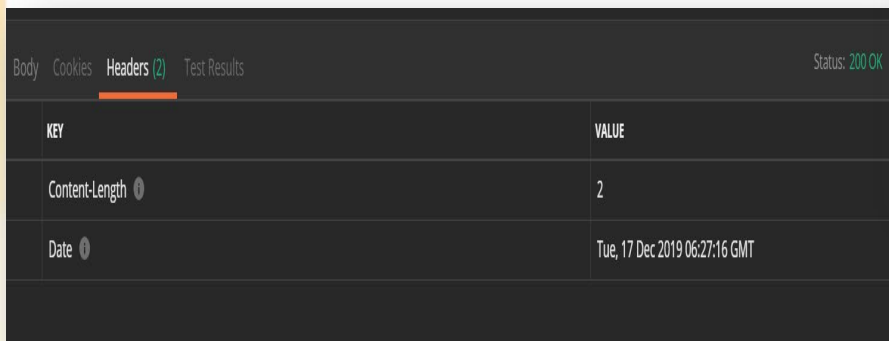
感測器 : DHT11\_1

2019-12-16T06:28:47 2019-12-17T06:28:48 50 筆 反序 查詢

搜尋時間範圍請使用 UTC 時間格式

數據顯示時間格式: UTC

感測數值(RAW)	Lat	Lon
2019-12-17T06:27:16Z {"temper":15.05,"humi":50.05}	25.053522	121.51707
2019-12-17T06:21:22Z {"temper":29.05,"humi":50.05}	25.053522	121.51707
2019-12-17T06:12:40Z {"temper":8.0,"humi":50.05}	25.053522	121.51707



# Lab

- ◆ 說明一下JavaBean設計架構與用途
- ◆ 使用JavaBean整合DAO設計一個客戶資料維護作業
- ◆ 如何將JavaBean序列化成JSON文件格式
- ◆ JSON文件格式整合在REST Service應用架構為何

客戶資料維護

客戶編號  
900

Frist Name  
Eric

Last Name  
Chen

EMAIL  
chen@cht.com.tw

傳送



```
true  
客戶編號:900  
First Name:Eric  
Last Name:Chen  
EMAIL:chen@cht.com.tw
```