**USBeacon 使用流程**

**後台 (**[**http://www.USBeacon.com.tw/console/index.html**](http://www.USBeacon.com.tw/console/index.html)**)**

1. 建立帳號, 點擊 立即註冊 後, 建立新的帳號.

2. 建立完成後, 會看到 Beacon UUID ([註1](#註1)) 和 Data Query UUID ([註2](#註2)).

3. 接著建立 USBeacon Data.

在 Server 端建立好的 USBeacon 資料, 是沒有 mac address ([註3](#註3)) 的, 在手機端的 config app 如果將這些資料燒錄至實體的 USBeacon, 然後上傳到 Server, 這樣每一筆 USBeacon 才會包含 mac address.

註1: Beacon UUID 這是將來要燒進 USBeacon 裡, 並連同 Major, Minor 一起廣播出來的 UUID.

註2: Data Query UUID, 這是在 app 中, 來向 Server 取得 Beacon List 所要使用的 UUID.

註3: Mac Address 是指每一個藍芽模組, 都會有一組獨一無二的 address, 這組 address 就叫作 Mac Address.

**USBeacon SDK**

SDK 主要分成兩大部份:

iBeaconScanManager 和 USBeaconConnection.

**iBeaconScanManager**

iBeaconScanManager 是透過 BluetoothAdapter 來 scan 附近的 Bluetooth device.

並分析 scan 到的 Bluetooth device 是否為 iBeacon.

若是的話, 則會呼叫 iBeaconScanManager.OniBeaconScan 的 onScaned, 並傳入 iBeaconData 物件.

這部份可以直接以 BluetoothAdapter#startLeScan() 來實作, 並不一定需要使用 iBeaconScanManager.

**USBeaconConnection**

這個物件主要是用來和 USBeacon Server 連線, 並取得編輯好的 USBeacon List.

在使用之前, 必須先將 Server 的資訊傳入 USBeaconConnection, 包括 server url, data query uuid 等.

在 後台 建立帳號後, 就能取得 data query uuid, 因為這組 UUID 是不會變動的, 所以可以直接寫進 App 當中.

USBeaconConnection#setServerInfo(USBeaconServerInfo serverInfo, OnResponse response) 就是用來設定 Server information 和 後台回應時的 call back method.

目前 Sample code 裡所使用的 data query UUID 是測試用的 UUID, 裡面記錄了太和光的資訊.

然後再透過 USBeaconConnection#checkForUpdates() 來確認 Server 的資料是否有更新.

若有的話, 則可呼叫 USBeaconConnection#downloadBeaconListFile() 來更新 app 的 USBeaconList.

若沒有的話, 則可呼叫 USBeaconConnection#loadLocalBeaconList() 來載入先前下載過的 USBeaconList 檔案.

App 將 scan 到的 iBeaconData 和 Server 端的 USBeaconData 來比較 UUID, Major, Minor, 檢查是否是相同的資料, 若是, 則 App 可顯示出 USBeaconData 所紀錄的 url, image, message ... 等.

iBeaconData#equals(iBeaconData iBeacon, boolean checkMacAddress) 就是做比較的功能.

另外, 若您有多組 USBeacon 的 UUID, Major, Minor 都燒成同樣的資料的話, 也可以透過不同的 mac address 來分辨是否有顯示不同的資料.

若您了解 Mac Address 的用法的話, 可以在做比較時, 將 checkMacAddress 設為 true, 不然請將它設成 false.

在使用 USBeacon 時, 其中最重要的就是 USBeaconData 這個類別了.

USBeaconData 包含了 4個不同距離的資料, 每一個距離除了 url, image 等以外, 還包含了使用者自定義資料.

USBeaconData 這個類別裡的 members 都是對映到 server 的每個欄位上. Sample code 裡已經展示了如何從 server 取得 USBeaconList 的流程, 至於這些欄位要如何呈現至 App 上, 就看這些欄位的用途是什麼了.

在 Demo App 裡, 目前是以太和光辦公室作為室內導航用, 可以至 google play 上下載這個 [Demo App](https://play.google.com/store/apps/details?id=com.THLight.USBeacon.Demo).

以下是幾個主要儲存資料的類別內容.

/\*\* scan iBeacon 時會回傳這個物件.\*/

public class iBeaconData

{

public String macAddress = "";

public UUID beaconUuid = UUID.*fromString*("00000000-0000-0000-0000-000000000000");

public int major = 0;

public int minor = 0;

/\*\* fixed rssi, broadcast from iBeacon. \*/

public byte oneMeterRssi = 0;

/\*\* real rssi analyzed by system. \*/

public byte rssi = 0;

}

/\*\* 從 server 下載 USBeaconList.zip 後, USBeaconConnection 和 XmlParserUSBeaconList 負責 unzip 和 parse 這個 zip file. 然後就可以從 USBeaconConnection 取得這個物件的 list 了. \*/

public class USBeaconData extends iBeaconData

{

/\*\* Distance enum. \*/

public enum Dist

{

*Immediate*,

*Near*,

*Far*,

*VeryFar*,

}

/\*\* unique id, USBeacon Server will generate an unique id for every USBeacon. \*/

public long id = (-1);

/\*\* group name, currently not used. \*/

public String group = "";

/\*\* USBeacon name. \*/

public String name = "";

/\*\* information in four distance range. \*/

public Map<Dist, USBeaconDistData> DistData= new LinkedHashMap<Dist, USBeaconDistData>();

}

/\*\* 不同距離的資料. \*/

public class USBeaconDistData

{

public String description = "";

public String phone = "";

public String strWebUrl = "";

public String strImageUrl = "";

public Image image = null;

public String remark = "";

public String youtube = "";

public List<USBeaconExtraColumn> extraData= new ArrayList<USBeaconExtraColumn>();

}

/\*\* 使用者自定義資料 \*/

public class USBeaconExtraColumn

{

public String title;

public String data;

}