行動裝置程式設計

Unit 12 傳統藍牙
Bluetooth Classic

蘇維宗 (Wei-Tsung Su)
suwt@au.edu.tw
564D

藍牙(Bluetooth)簡介

藍牙是一種無線個人網路(Personal Area Network)技術

- 傳統藍牙(Bluetooth Classic)適用於有高頻寬需求的應用(如多媒體)
- 藍牙4.0以後併入Bluetooth Low Energy (BLE)。

Bluetooth Special Interest Group (SIG)提出藍牙標準與特定應用的Profiles以提高相容性。Profile指的是特定藍牙應用的規格，例如Headset (藍芽耳機)、Advanced Audio Distribution Profile (A2DP)、Health Device Profile (HDP)等。

藍牙(Bluetooth)簡介(續)

每個藍牙服務必須要有一個獨一無二的Universally Unique ID (UUID)，以利於搜尋透定的藍牙服務。

UUID的長度為128個位元，用途類似TCP/IP中的通訊埠(port)，例如

- Human Interface Device Profile (HID)的UUID為0x0011
  - 00000011-0000-1000-8000-00805F9B34FB
- Serial Port Profile (SPP)的UUID為0x1101
  - 00001101-0000-1000-8000-00805F9B34FB
- ...

開發傳統藍牙應用程式

- 取得藍牙的使用權限
- 確定裝置支援藍牙
- 確定裝置開啟藍牙
- 取得藍牙裝置清單
  - 取得已配對裝置
  - 搜尋附近藍牙裝置
  - 啟用可被搜尋
- 連結藍牙裝置
  - 連結非依據Profile設計的藍牙裝置
    - 利用BluetoothServerSocket類別實作Server端 或
    - 利用BluetoothSocket類別實作Client端
  - 連結依據Profile設計藍牙裝置
    - 利用實作BluetoothProfile介面的類別

取得藍牙的使用權限

在Manifest檔中根據需求加入下列藍牙的使用權限

- android.permission.BLUETOOTH
  - 允許連結已配對的藍牙裝置
- android.permission.BLUETOOTH_ADMIN
  - 允許搜尋並配對的藍牙裝置

確定裝置支援藍牙

利用BluetoothManager與BluetoothAdapter類別與藍牙介面溝通

```
1.   private BluetoothManager mBtManager;
2.   private BluetoothAdapter mBtAdapter;
3.   ...
4.   mBtManager = (BluetoothManager)getSystemService(Context.BLUETOOTH_SERVICE);
5.   mBtAdapter = mBtManager.getAdapter();
6.   if (mBtAdapter == null) {
7.     // 此裝置不支援藍牙
8.   } else {
9.     // 此裝置支援藍芽
10.  }
```

確定裝置開啟藍牙

利用BluetoothAdapter類別確定藍牙介面是否開啟

```
1.  final int REQUEST_ENABLE_BT = 0;
2.  ...
3.  if (!mBtAdapter.isEnabled()) { // 藍牙未開啟，詢問使用者是否開啟
4.    Intent intent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
5.    startActivityForResult(intent, REQUEST_ENABLE_BT);
6.  } else {
7.    // 藍牙已開啟
8.  }
```

確定裝置開啟藍牙(續)

在onActivityResult() 中根據使用者的決定來做不同的回應

```
1.   @Override
2.   protected void onActivityResult(int requestCode, int resultCode, Intent data){
3.     super.onActivityResult(requestCode, resultCode, data);
4.     if(requestCode == REQUEST_ENABLE_BT) {
5.       switch(resultCode) {
6.       case RESULT_OK:        //  使用者選擇開啟藍牙
7.         break;
8.       case RESULT_CANCELED: //  使用者選擇不開啟藍牙
9.         break;
10.      }
11.    }
12.  }
```

取得已配對裝置

如果裝置已經配對過，可以直接取得已配對裝置的清單

```
1.  Set<BluetoothDevice> pairedDevs;
2.  pairedDevs = mBtAdapter.getBondedDevices();
3.  if(pairedDevs.size() > 0) {
4.    for (BluetoothDevice device : pairedDevs) {
5.      // Do something on each paired device
6.    }
7.  }
```

搜尋附近藍牙裝置

開始(停止)搜尋附近的藍牙裝置

```
1.  mBtAdapter.startDiscovery();    //  開始搜尋
2.  ...
3.  mBtAdapter.cancelDiscovery();   //  停止搜尋(省電)
4.  ...
```

搜尋開始時系統會發出 **BluetoothAdapter.ACTION_DISCOVERY_STARTED**廣播。
搜尋結束時系統會發出 **BluetoothAdapter.ACTION_DISCOVERY_FINISHED**廣播。
搜尋到裝置時系統會發出 **BluetoothDevice.ACTION_FOUND**廣播。

搜尋附近藍牙裝置(續)

實作BrocastReceiver接收BluetoothDevice.ACTION_FOUND廣播

```
1.   private final BroadcastReceiver mReceiver = new BroadcastReceiver() {
2.     public void onReceive(Context context, Intent intent) {
3.       String action = intent.getAction();
4.       if(action.equals(BluetoothDevice.ACTION_FOUND.equals)) {
5.         BluetoothDevice device =
6.             intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
7.         // Do something on the discovered device
8.       }
9.     }
10.  };
```

搜尋附近藍牙裝置(續)

實作BrocastReceiver接收BluetoothDevice.ACTION_FOUND廣播

```
11.   ...
12.   IntentFilter filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
13.   registerReceiver(mReceiver, filter);
14.   ...
15.   unregisterReceiver(mReceiver);
```

啟用可被搜尋

在程式中開啟可被搜尋的功能

```
1.  final int REQUEST_ENABLE_DISCOVERABLE = 1;
2.  final int DURATION = 300;
3.  ...
4.  Intent intent = new Intent(BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE);
5.  intent.putExtra(BluetoothAdapter.EXTRA_DISCOVERABLE_DURATION, DURATION);
6.  startActivityForResult(intent, REQUEST_ENABLE_DISCOVERABLE);
```

啟用可被搜尋(續)

在onActivityResult() 中根據使用者的決定來做不同的回應

```
1.   @Override
2.   protected void onActivityResult(int requestCode, int resultCode, Intent data){
3.     super.onActivityResult(requestCode, resultCode, data);
4.     if(requestCode == REQUEST_ENABLE_DISCOVERABLE) {
5.       switch(resultCode) {
6.       case RESULT_OK:         //  使用者選擇允許被搜尋
7.         break;
8.       case RESULT_CANCELED:   //  使用者選擇不允許被搜尋
9.         break;
10.      }
11.    }
12.  }
```

課堂作業

試著實作一個`BroadcastReceiver`, 可以在藍牙搜尋裝置時接收以下廣播並顯示Toast訊息。

- `BluetoothDevice.ACTION_FOUND`
  - 搜尋到藍牙(請顯示裝置名稱與位址)
- `BluetoothAdapter.ACTION_DISCOVERY_STARTED`
  - 藍牙搜尋已開始
- `BluetoothAdapter.ACTION_DISCOVERY_FINISHED`
  - 藍牙搜尋已結束

Q & A

Computer History Museum, Mt. View, CA