



# 行動裝置程式設計

## Unit 10 資料庫 SQLite Database

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





## 儲存結構化資料

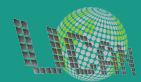
當Apps需要處理大量的結構化資料(structured data)時，可以選擇將資料儲存在本機端的資料中。(另一個選擇是可以儲存在雲端伺服器上。)

在Android中，可選擇以下兩種方式來存取本機端資料庫

1. 使用[SQLite API](#)直接存取SQLite
2. 使用[Room](#)間接存取SQLite

---

# SQLite簡介





# SQLite

- SQLite是一個適用於嵌入式裝置的小型資料庫管理系統
  - Android與iPhone皆採用SQLite
- SQLite官方網站
  - <http://www.sqlite.org/>
- SQLite語法結構
  - <http://www.sqlite.org/lang.html>

# SQLite基本語法(建立資料表)

## SQL語法

**CREATE TABLE** 資料表名稱  
(欄位1名稱 欄位1型態 欄位1特性,  
欄位2名稱 欄位2型態 欄位2特性,  
...  
欄位n名稱 欄位n型態 欄位n特性);

## 欄位型態, 例如

**INTEGER** 整數  
**TEXT** 文字  
**REAL** 實數  
...

## 欄位特性, 例如

**PRIMARY KEY**表示該欄位為KEY  
**NOT NULL**表示該欄位不能空白  
**AUTOINCREMENT**表示會自動產生流水號

...





# SQLite基本語法(建立表格)

## SQL語法範例

```
CREATE TABLE IF NOT EXISTS bmi
(id INTEGER PRIMARY KEY AUTOINCREMENT,
 name TEXT NOT NULL,
 weight REAL NOT NULL,
 height REAL NOT NULL,
 year INTEGER NOT NULL,
 month INTEGER,
 day INTEGER);
```

bmi						
id	name	weight	height	year	month	day

# SQLite基本語法(新增資料)

## SQL語法

INSERT INTO 資料表名稱 [(欄位名稱,...)] VALUES (欄位值,...);

## SQL語法範例

```
INSERT INTO bmi VALUES (1, '小明', 70, 1.6, 2010, 5, 6);
```

```
INSERT INTO bmi (name, weight, height, year, month, day)  
VALUES ('小華', 60, 1.7, 2010, 5, 8);
```

bmi						
id	name	weight	height	year	month	day
1	小明	70	1.6	2010	5	6
2	小華	60	1.7	2010	5	8



# SQLite基本語法(更新資料)

## SQL語法

UPDATE 資料表名稱 SET 欄位名稱=值 WHERE 條件;

## SQL語法範例

UPDATE bmi SET day=7 WHERE name=' 小明 ';

bmi						
id	name	weight	height	year	month	day
1	小明	70	1.6	2010	5	7
2	小華	60	1.7	2010	5	8



# SQLite基本語法(刪除資料)

## SQL語法

DELETE FROM 資料表名稱 WHERE 條件;

## SQL語法範例

DELETE FROM bmi WHERE name='小明';

bmi						
id	name	weight	height	year	month	day
<del>1</del>	<del>小明</del>	<del>70</del>	<del>1.6</del>	<del>2010</del>	<del>5</del>	<del>7</del>
2	小華	60	1.7	2010	5	8

# SQLite基本語法(查詢資料)

bmi						
id	name	weight	height	year	month	day
2	小華	60	1.7	2010	5	8

## SQL語法

SELECT 指定欄位列表 FROM 資料表名稱 WHERE 條件;

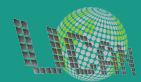
## SQL語法範例

SELECT \* FROM bmi WHERE name='小華';

SELECT name,weight FROM bmi WHERE name='小華';

---

# SQLite基本操作





# 使用adb練習操作SQLite

透過Android Debug Bridge (adb)可以終端機方式登入系統來進行操作

- 開啟命令列模式執行adb shell連接AVD
  - `# cd data/data/[package name]`
- 建立資料庫
  - `# mkdir databases`
  - `# cd databases`
  - `# sqlite3 [資料庫名稱]`
- 顯示目前建立的資料庫
  - `sqlite> .databases`





## 使用adb練習操作SQLite (續)

- 建立資料表

- `sqlite> CREATE TABLE IF NOT EXISTS schedule`  
`...>(id INTEGER PRIMARY KEY AUTOINCREMENT,`  
`...> date TEXT NOT NULL,`  
`...> event TEXT NOT NULL);`

- 顯示目前資料庫中的所有資料表

- `sqlite> .tables`

- 顯示特定資料表的schema

- `sqlite> .schema schedule`

- 離開SQLite

- `sqlite> .quit`



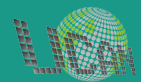


## SQLite基本操作(建立資料表)

- 請每次執行完下述動作後利用SQL語法查詢資料
  - 查詢event為國慶日的資料
  - 查詢並只顯示date與event欄位
- 請利用SQL語法新增兩筆資料到schedule資料表中(日期格式: YYYY-MM-DD HH:MM:SS)
  - 2012年10月10日早上7點00分00秒國慶日
  - 2012年12月11日下午2點30分00秒學術演講
- 請利用SQL語法將國慶日的date改成2012年10月10日早上6點00分00秒
- 請利用SQL語法刪除國慶日資料

---

# 使用SQLite API直接存取SQLite






# 開啟/關閉資料庫

使用[SQLiteDatabase](#)類別開啟或產生資料庫

```
1. private SQLiteDatabase db;  
2. //開啟或建立資料庫  
3. db = openOrCreateDatabase ([資料庫名稱], MODE_PRIVATE, null);  
4. ...  
5. //關閉資料庫  
6. db.close();
```







## 執行SQL語法(無回傳資料)

當執行的SQL語法無回傳資料時(例如, 開啟(新增)資料表與新增/刪除/更新資料等), 可直接呼叫SQLiteDatabase類別的[`execSQL\(String SQL\)`](#)方法

1. //準備好要執行的SQL語法
2. `String sql = "DELETE FROM schedule WHERE event=\ ' 國慶日 \ '";`
3. `db.execSQL(String SQL);`



## 執行SQL語法(有回傳資料)

當執行的SQL語法有回傳資料時，需呼叫SQLiteDatabase類別的[rawQuery\(\)](#)方法

```
1. // 準備好SELECT語法
2. String sql = "SELECT * FROM schedule WHERE event='\''國慶日'\''";
3. Cursor cc = db.rawQuery(String sql, null); // 預設cc會指到第1筆資料之前
4. // 移動指向資料的指標
5. cc.moveToFirst(); //移到第一筆資料
6. cc.moveToLast(); //移到最後一筆資料
7. cc.moveToPrevious(); //移到前一筆資料
8. cc.moveToNext(); //移到後一筆資料
9. ...
```



## 執行SQL語法(有回傳資料)

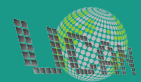
當執行的SQL語法有回傳資料時，需呼叫SQLiteDatabase類別的[rawQuery\(\)](#)方法

```
1. // 準備好SELECT語法
2. String sql = "SELECT * FROM schedule WHERE event='\''國慶日'\''"
3. Cursor cc = db.rawQuery(String sql, null);
4. ...
5. // 取出目前資料中指定欄位 (columnIndex) 的資料
6. cc.getInt(int ColumnIndex); // id欄位index為0且型別為整數
7. cc.getString(int ColumnIndex); // date欄位index為1且型別為字串
8. cc.getString(int ColumnIndex); // event欄位index為2且型別為字串
```



---

# 使用ListView呈現資料庫資料



# 建立新的Layout

使用預設的LinearLayout

加入ListView元件(假設id為list)

ListView元件常用於條列性質相同的資料，例如

- 電子郵件
- 聯絡人
- 簡訊
- ...





## 在ListView上顯示資料列表

利用ListAdapter類別產生顯示於

```
1.  ListView list;
2.  ArrayList<String> data = new ArrayList<String>();    //可動態產生資料
3.  ...
4.  list = findViewById(R.id.list);
5.  data.add("資料A");                                     //動態產生1筆資料
6.  ListAdapter adapter = new ArrayAdapter<String>(
                        this, android.R.layout.simple_list_item_1, data);
7.  list.setAdapter(adapter);
```





## 使用ContextMenu處理資料(續)

註冊ContextMenu (在ListView上)

```
1. registerForContextMenu(list);
```

產生ContextMenu

```
1. @Override
2. public void onCreateContextMenu(ContextMenu menu, View v, ContextMenuInfo info) {
3.     super.onCreateContextMenu(menu, v, menuInfo);
4.     menu.setHeaderTitle("動作");
5.     menu.add(0, 0, 0, "編輯");
6.     menu.add(0, 1, 1, "刪除");
7. }
```





# 使用ContextMenu處理資料

產生ContextMenu事件處理函式

```
1.  @Override
2.  public boolean onContextItemSelected(MenuItem item) {
3.      //取得資料位置
4.      AdapterContextMenuInfo info = (AdapterContextMenuInfo) item.getMenuInfo();
5.      String listItem = data.get(info.position);
6.      switch(item.getItemId()) {
7.          case 0: // 執行編輯功能
8.              break;
9.              ...
10.         }
11.         return super.onContextItemSelected(item);
12.     }
```







# 作業

1. 利用資料庫製作簡易行事曆
2. 利用ListView來顯示行事曆內容
3. 利用ContextMenu來刪除/編輯行事曆項目
4. 利用OptionsMenu新增行事曆項目

# Q & A

---



Computer History Museum, Mt. View, CA

