**Lab11**

**SQLite**

本節目的：

* 介紹SQLite用途。
* 建立SQLite資料庫，並對資料庫裡資料表做**新增**、**修改**、**刪除**和**查詢**的基本操作。
  1. **觀念說明：**

SQLite是一個由C語言撰寫的小型關聯式資料庫管理系統，與一般資料庫不同在於它不是一個主從關係結構的資料庫，而是被整合在應用程式中的嵌入式資料庫。Android 應用程式可以將資料儲存在手機上SQLite中，作為資料的快取之用，缺點是本地資料庫與伺服器的資料會有不同步的疑慮。

如下圖 1所示， Chrome APP使用SQLite資料庫儲存Cookies、Favicons與History…等使用資料。

|  |  |
| --- | --- |
| 圖 1-1、App列表 | 圖 1-2、資料庫列表 |

舉凡Line、Facebook…等應用程式，皆有使用到資料庫儲存如個人設定、聊天訊息….等大量使用資料，並且可以作為資料快取之用。

說明

* + 1. **建立SQLiteOpenHelper**

Android 提供「android.database.sqlite」套件，可以處理資料庫的工作。在這個套件中的「SQLiteOpenHelper」類別，能夠讓應用程式執行建立資料庫和表格等，因此第一步我們要先建立一個SQLiteOpenHelper的物件。

首先圖 2選擇File/New/Java Class。

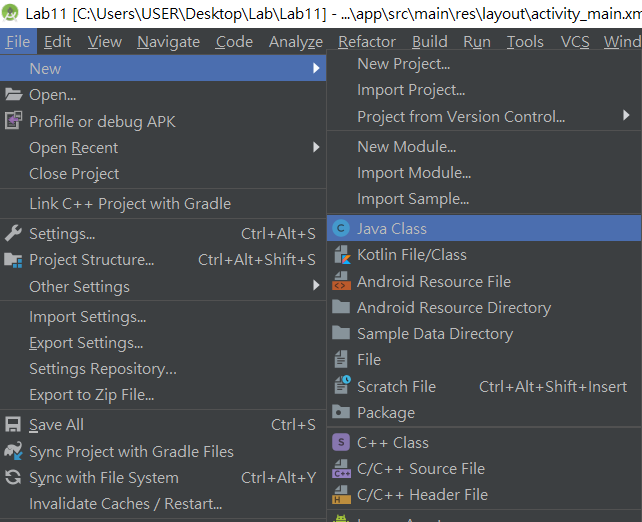


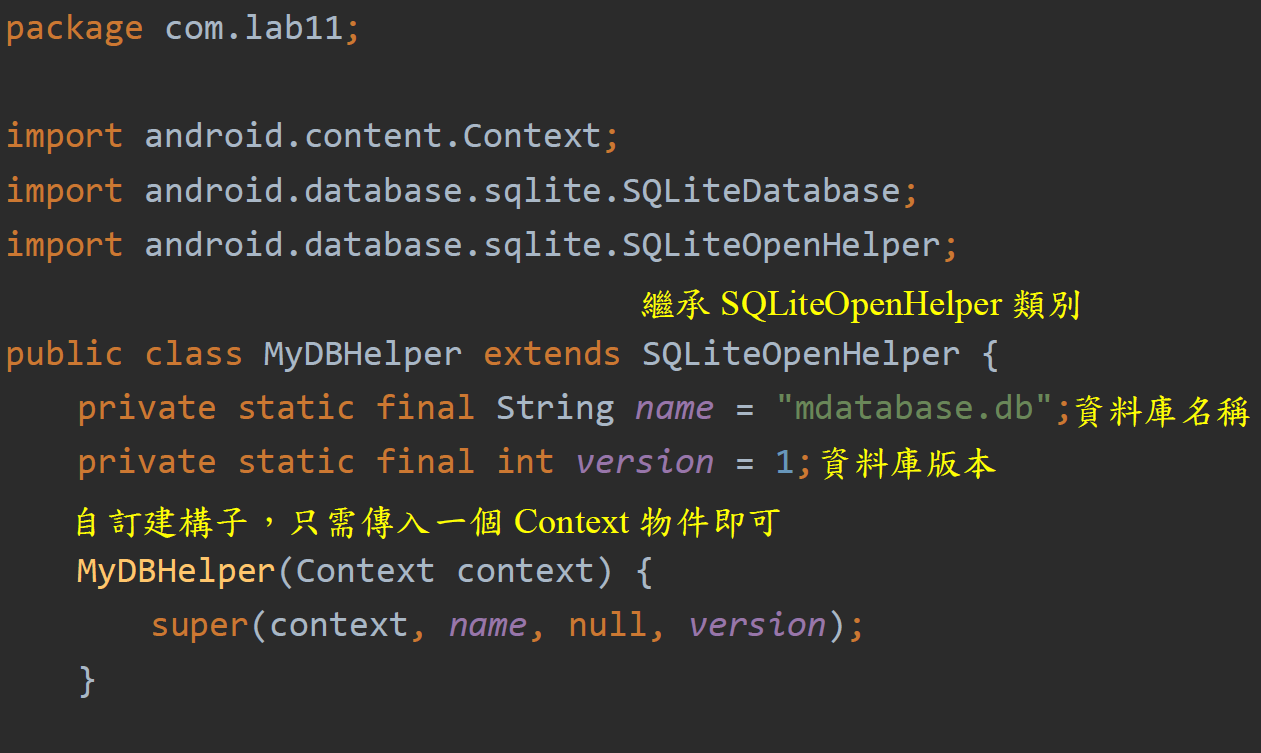
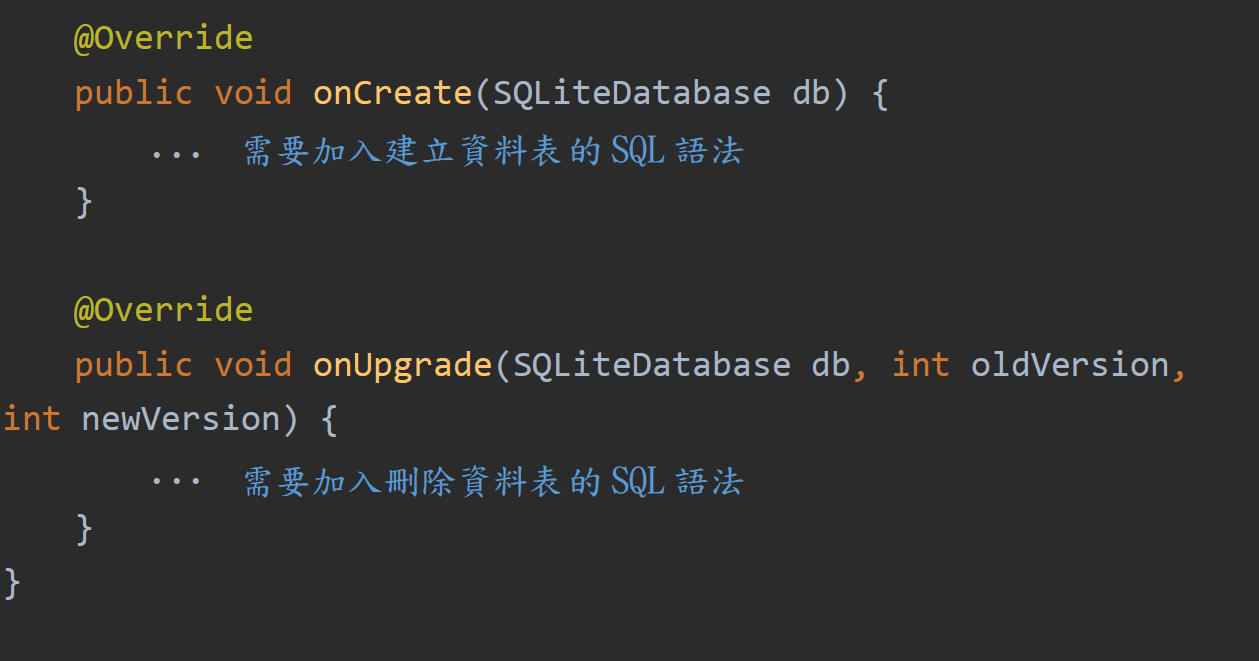
圖 2、產生新的Class

圖 3在Name輸入「MyDBHelper」並選擇「**OK**」



圖 3、創建MyDBHelper

這步驟中會產生出一個名為MyDBHelper的空白Class檔，而我們要繼承自SQLiteOpenHelper來使用其功能，因此修改加入語法如下：

應用程式第一次在裝置執行的時候，由SQLiteOpenHelper負責建立需要的功能，而之後執行的時候會使用已經建立好的資料庫。

* + 1. **設計資料庫表格**

SQLite是資料庫（Database），因此要先了解原理。資料庫代表應用程式儲存和管理資料的單位，應用程式透過資料庫來存取不同的資料。一個資料庫通常擁有數個資料表，下圖 4的資料庫中有乘客、司機與訂單三種資料表，分別存放三種不同類型的資料。

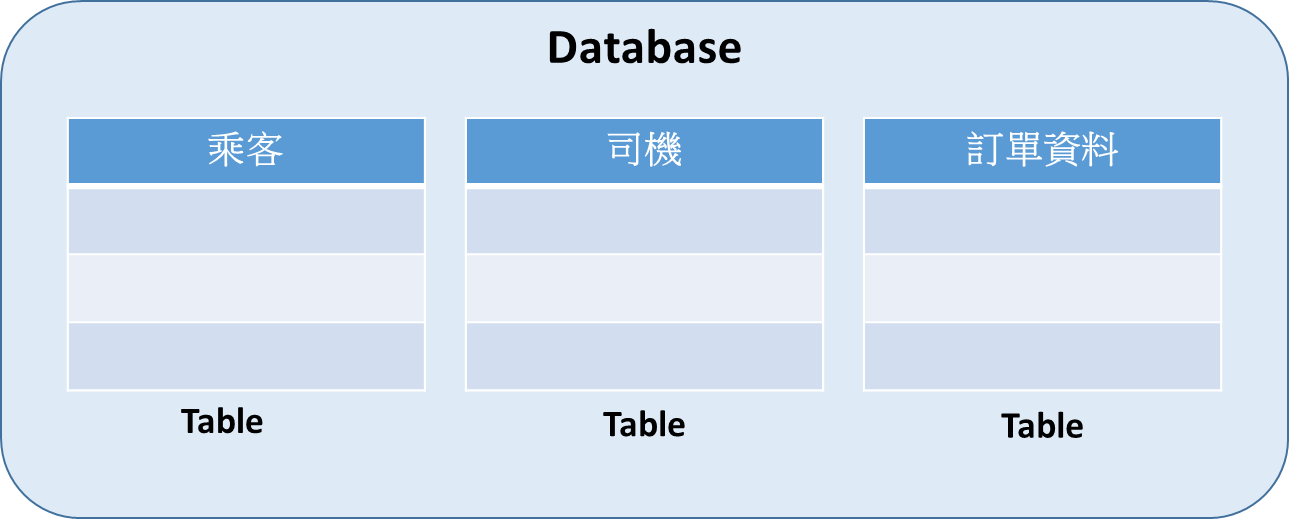


圖 4、資料庫與資料表示意圖

例如一個搭車的資料庫，就需要儲存與管理乘客、司機和訂單資料。每一種定義在資料庫中的資料稱為表格（Table），例如乘客表格可以儲存所有的乘客資料。

SQLite 資料庫必須先建立好資料庫與表格後，才可以執行存取與資料管理的工作。

建立**資料庫表格**使用SQL的「CREATE TABLE」指令，這個指令需要指定表格的名稱，還有這個表格用來儲存每一筆資料的欄位（Column）。例如以下指令會產生出一個名為myTable的表格：

*注：SQL語法中沒有大小寫之分。*

**CREATE TABLE myTable()**

表格最後面的括弧中我們要加入表格欄位的語法，每個資料庫表格中可以放入數個表格欄位，在設計表格欄位的時候，需要設定欄位名稱和型態，型態如int、String等會決定這欄位能夠儲存何種類型的變數，不過SQLite資料庫的資料型態只有下面三種，透過它們來決定表格欄位儲存的資料型態：

* **INTEGER**整數，對應到byte、short、int 和long。
* **REAL**小數，對應到的float 和double。
* **TEXT**字串，對應到String。

實現後的語法如下：

**TITLE INTEGER**

紅色的字表示欄位名稱，而綠色的字表示變數型態。

通常在欄位中還會新增「NOT NULL」的指令，表示這個欄位不允許空值，可以避免許多資料發生問題。

**TITLE INTEGER NOT NULL**

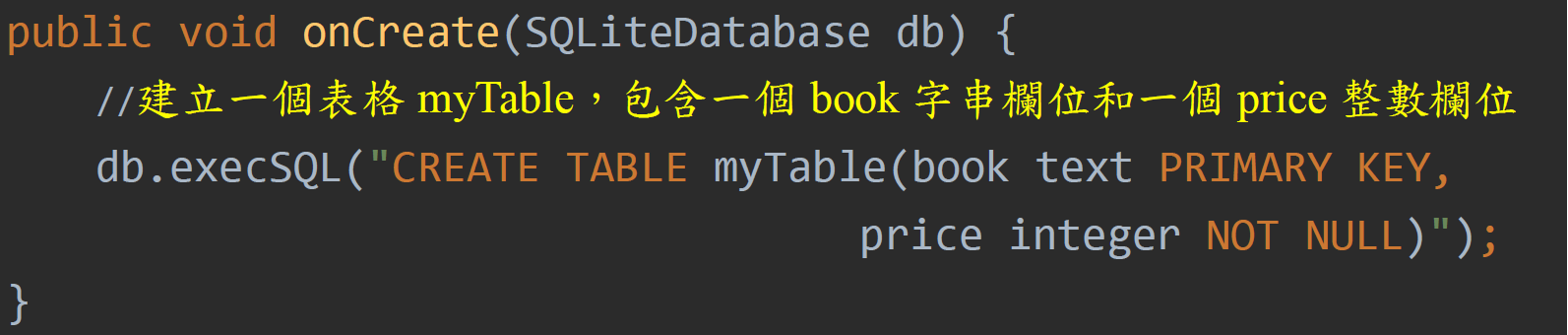
此外一個資料表必須包含一個[主鍵]欄位，這個欄位必須是唯一的值，用於索引每一筆新產生出來的資料，因此SQLite表格建議要包含一個欄位名稱內容唯一的主鍵、後面加上「PRIMARY KEY」的欄位。

**book TEXT PRIMARY KEY**

結合以上語法，我們假設要創建一個名為myTable，有一個book(String)、price(Integer)的欄位，我們編寫之後的字串如下：

**CREATE TABLE myTable(book TEXT PRIMARY KEY, price INTEGER NOT NULL)**

此即為創建表單的SQL語法，而SQLite中我們要在MyDBHelper裡的onCreate(SQLiteDatabase db)中將此語法字串傳入以產生出表單。

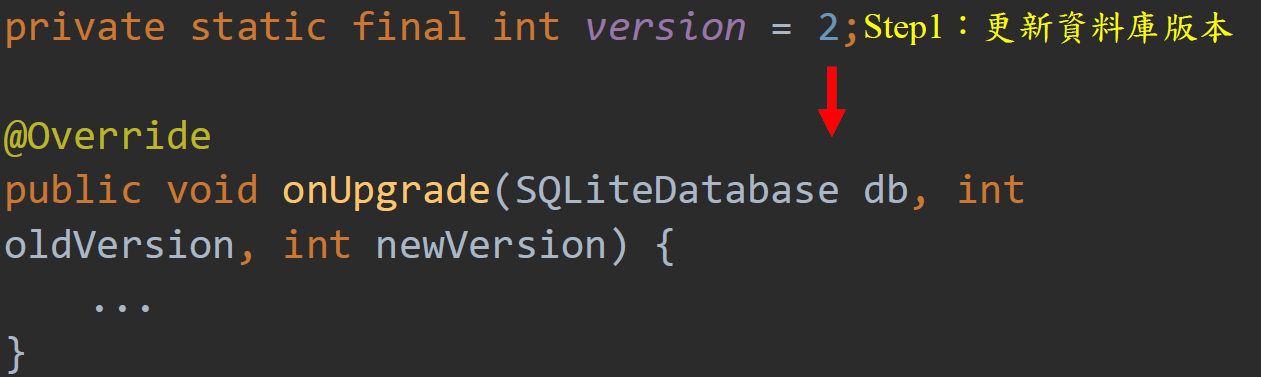


onCreate(SQLiteDatabase db)只會在創建資料表時執行，之後便不再執行，如果想要更新資料表的欄位，就需要重建資料庫。重建的流程如下圖 5：

圖 5、重建資料庫流程

重建資料庫需要有三個步驟。

[**Step1**]必須要修改資料庫版本，SQLiteOpenHelper偵測到資料庫版本更新時，會調用onUpgrade()方法，而我們需要利用onUpgrade()來做刪除表格的工作。

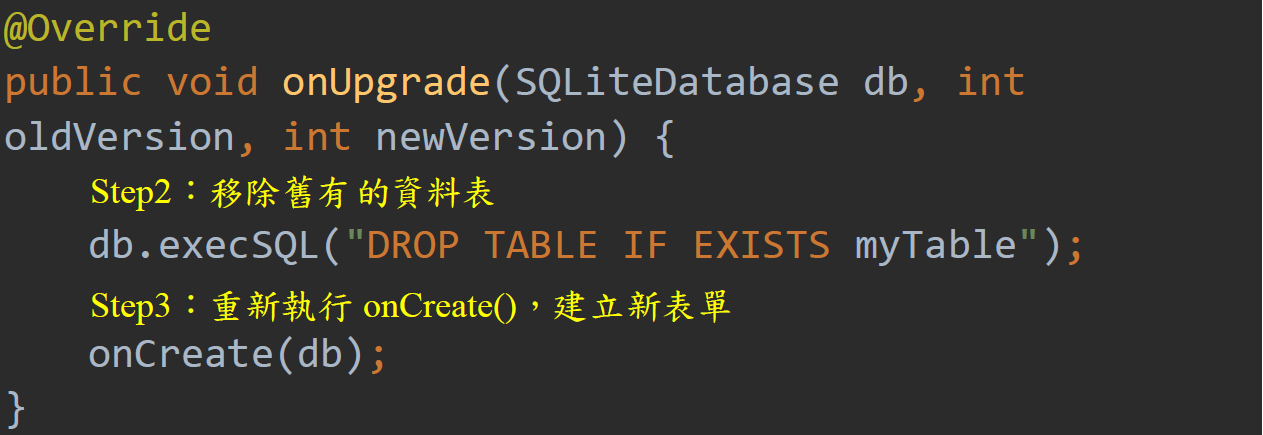


[**Step2**]在onUpgrade()中我們要加入一段SQL語法來刪除表格

**DROP TABLE IF EXISTS myTable**

我們使用「DROP TABLE IF EXISTS」實現刪除指定的的動作，這指令意思是如果myTable已經存在則將其刪除。

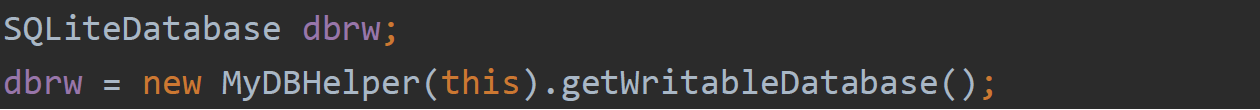
onUpgrade修改後如下：



[**Step3**]刪除資料表之後，需要再次呼叫onCreate()來建立新的資料表。

* + 1. **使用資料庫**

完成了建置資料庫的前置動作之後，下一步我們要實際的在程式去使用設計好的資料庫。



一開始我們需要產生MyDBHelper的物件實體，並且透過getWritableDatabase()來建立起SQLiteDatabase類別，而SQLiteDatabase就是我們的資料庫本體，後續的新增、查詢、修改、刪除資料功能都需要使用這個物件。

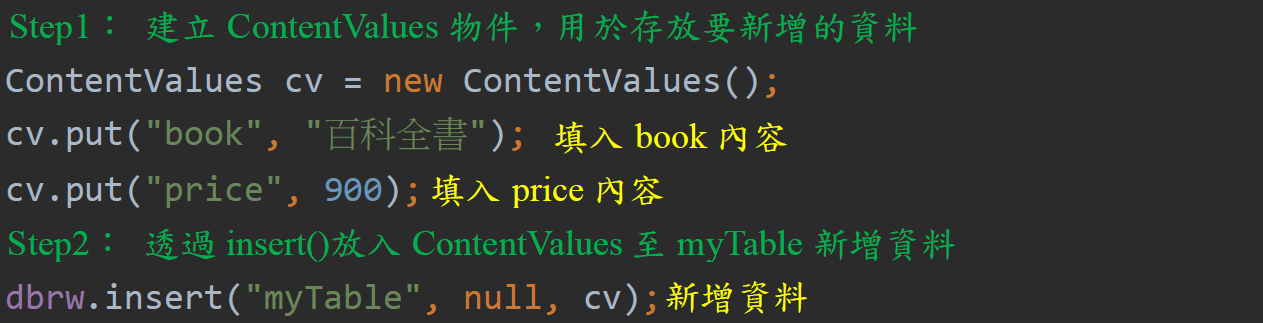
* 新增資料

前面我們創建了一個myTable的資料表，要增加一筆資料，下圖 6新增百科全書：



圖 6、新增百科全書至資料表

而對應的語法如下：



這段語法中會新增一筆百科全書、價格為900的資料，這邊需要使用到一個ContentValues物件，因為一個欄位名稱(key)會對應到一筆資料內容(value)，我們需要存放資料存到對應的欄位名稱之下，因此ContentValues能幫我們包裝資料。

我們透過ContentValues分別對兩個表格填入資料，第一個參數要放入欄位名稱，第二個參數要放入資料內容。之後在使用SQLiteDatabase.insert語法將資料存放到myTable之中。另外，這邊要注意，如果資料內容的型態與前面訂定的資料欄位型態不同，是無法加入的。

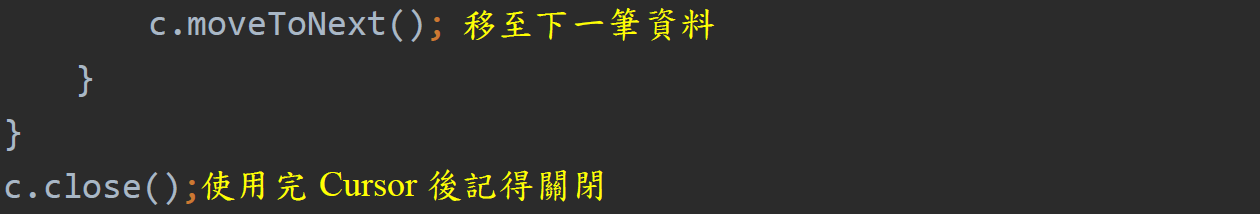
* 查詢資料

查詢是四種操作方式中最複雜的功能，如要查詢某些資料，如下圖 7從資料表中查詢百科全書：



圖 7、查詢資料表

程式中需要加入以下語法：



此段中我們使用SQLiteDatabase.query()的方法取得book為百科全書的資料，要查詢資料，我們需要提供**查詢條件**及**要取得的欄位**兩個重要參數：

* **查詢條件：**

要查詢某些資料時，我們需要告知要那些資料，例如查詢書籍，需要明確說明查詢的書名、類型等資訊。程式中的描述如下：

欄位名稱 = "資料內容"

資料庫篩選出來欄位名稱符合該筆資料內容的項目，如果沒有填入任何的條件(要填入null)，則會顯示所有資料。

* **要取得的欄位：**

查詢到資訊之後，資料庫可以不用回傳所有的欄位，我們可以限定只取得某些欄位，例如查書時可能只需要書名與價錢，這樣就可以減少不必要的資訊。

而要實現這功能，我們需要使用一組字串陣列，並填入想要回傳的資料欄位名稱，如下：

String[] colum = {"欄位名稱1", "欄位名稱2", "欄位名稱3"};

而SQLiteDatabase.query()會回傳一個Cursor類別的結果，Cursor可以想像程式一張資料表，篩選後的資料表如下圖 8：



圖 8、篩選後的資料表

Cursor.getCount()可以取得查詢到的總比數，我們可以使用這方法來確認是否有資料以及需要取幾次資料。

Cursor使用get(欄位順序)來依序取得資料內容，由於SQLite比較不嚴謹，如果目的是顯示資料可以都用Cursor.getString()來取值，而這個欄位順序等同於上面設定的colum。也就是說，如果colum = {"book", "price"}的話，Cursor.getString(0)可以取得book的資料內容，而Cursor.getString(1)可以取得price的資料內容。

要當要移動至其他筆資料時，Cursor提供一種非常簡單的方式移動。那就是每一次使用Cursor.moveToNext()來移動至下一筆項目，一直到資料的最後一筆為止，因此一開始需要使用Cursor.moveToFirst()移動到第一筆資料，以確保不會遺漏任何筆資料。

* **修改資料**

建立ContentValues物件

當某筆資料需要做修正，如下圖 9更正百科全書的價格為200



圖 9、更新資料庫的資料

我們會需要使用到SQLiteDatabase.update()的語法如下：



這段語法會先找出所有的book為百科全書的資料，並且將price=900的資料寫入進去，因此只要是book為百科全書的資料，其價格都會變成200。

* **刪除資料**

當某筆資料需要移除時，如下圖 10從資料庫中刪除百科全書的資料



圖 10、刪除資料表中的資料

我們可以使用SQLiteDatabase.delete()將其刪除。要實作的語法如下：



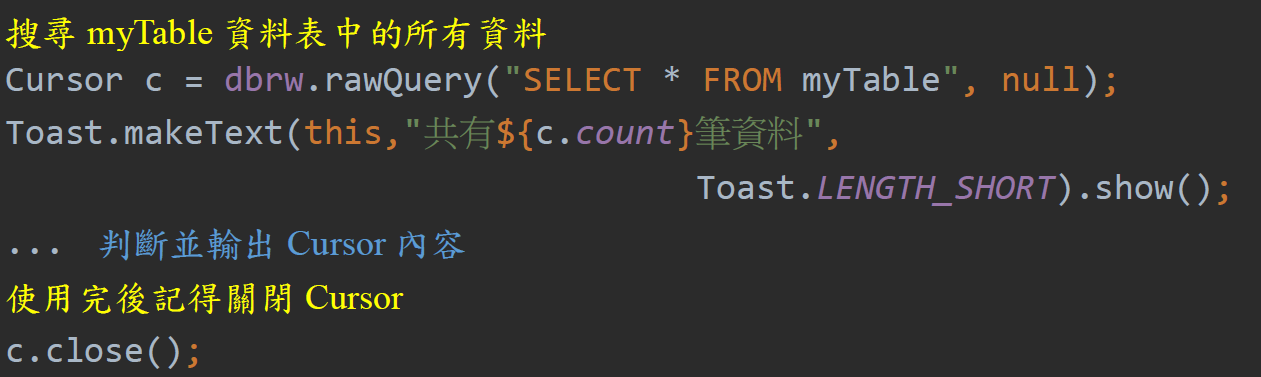
語法使用上與查詢類似，需要描述要查詢的資料為何，如此語法中會篩選出所有的book為百科全書的資料，並且將其刪除。

* + 1. **使用結構化查詢語言SQL**

除了使用SQLiteOpenHelper提供的基礎函式，SQLiteOpenHelper也支援直接使用結構化查詢語言(SQL)對資料庫進行管理，分為資料查詢與資料異動兩種使用方式。

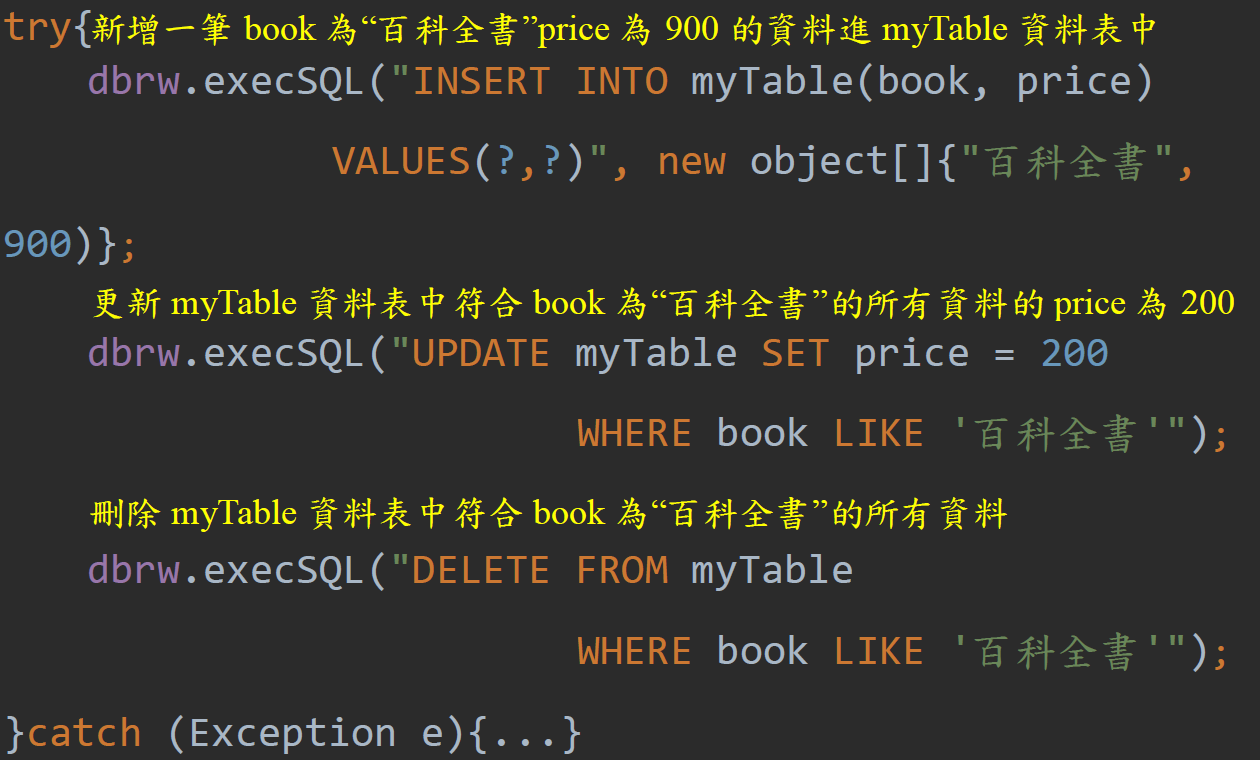
* **資料查詢**

當我們要查詢某筆資料時，可以使用SQLiteDatabase.rawQuery()的語法，與SQLiteDatabase.query()一樣會回傳一個Cursor類別的結果。



* **資料異動(新增、刪除、修改、取代...等)**

當我們要更動資料庫的資料時，可以使用SQLiteDatabase.execSQL()的語法，execSQL()並沒有任何回傳值，通常會搭配Try Catch一同使用，當指令成功時程式會繼續運作，而失敗時則會拋出Exception錯誤。



* 1. **設計重點：**
* 使用SQLite資料庫創建一個圖 11圖書管理系統，可以增加、查詢、修改、刪除書籍資訊(書名、價格)。



圖 11、圖書管理系統

* 圖 12輸入書名、價格後按下新增可以新增一本書

|  |  |
| --- | --- |
| 圖 12-1、新增百科全書 | 圖 12-2、新增成功 |

* 圖 13按下查詢可以列出所有的書，而如果有輸入書名，僅會列出符合書名的書

|  |  |
| --- | --- |
| 圖 13-1、查詢所有書籍 | 圖 13-2、查詢英文雜誌 |

* 圖 14輸入書名、價格後按下修改可以修改一本書的價格，再按下查詢後可以看到書本的價格被修改

|  |  |
| --- | --- |
| 圖 14-1、修改英文雜誌的價格 | 圖 14-2、查詢結果 |

* 圖 15輸入書名按下刪除可以刪除一本書，再按下查詢後可以看到書本被刪除

|  |  |
| --- | --- |
| 圖 15-1、刪除英文雜誌 | 圖 15-2、查詢結果 |

* 1. **設計步驟：**

Step1 建立新專案，以及下圖 16對應的class與xml檔：

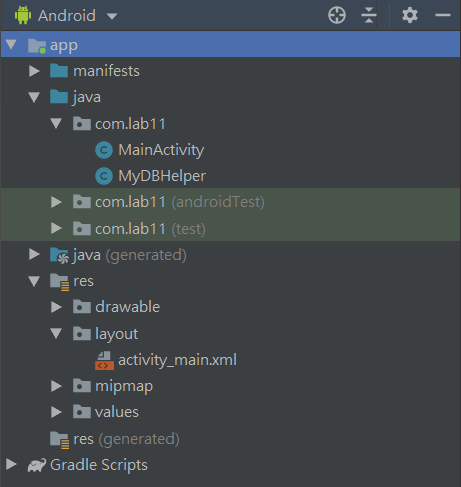
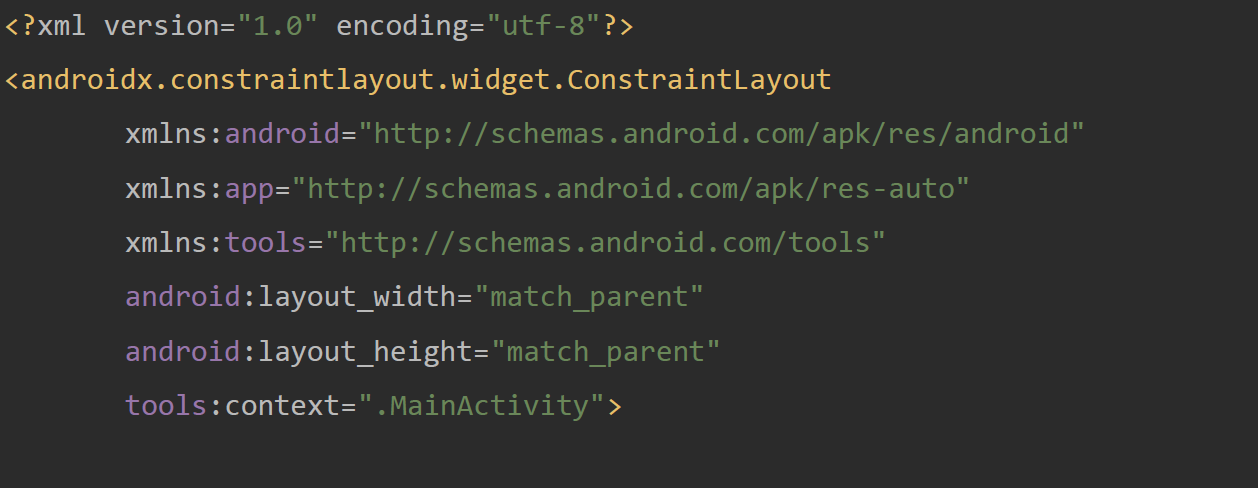


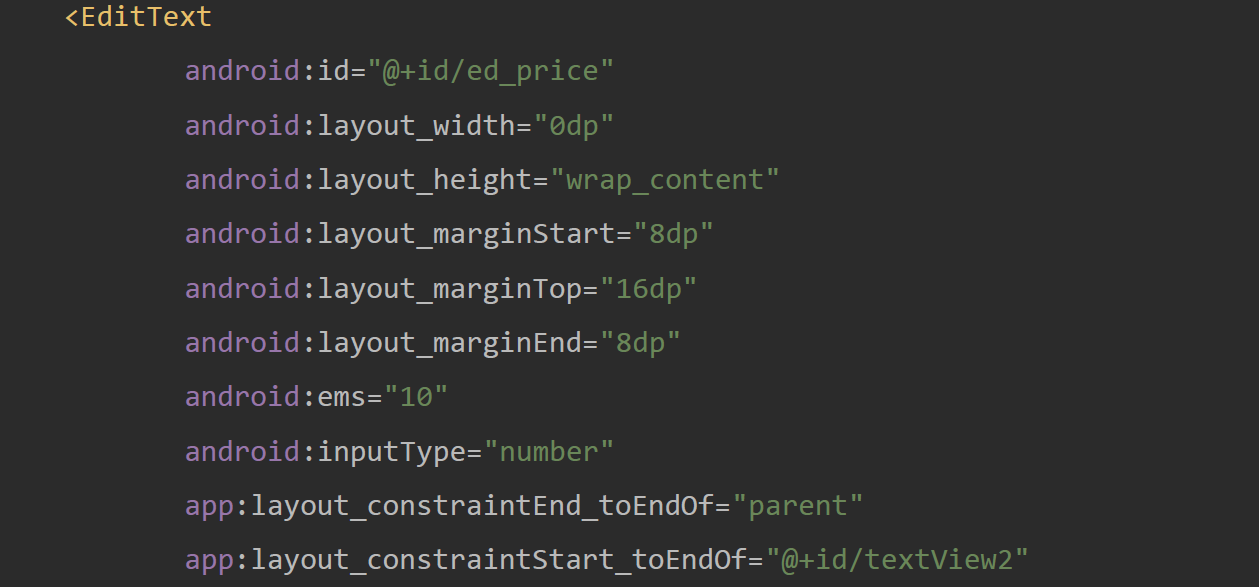
圖 16、專案架構

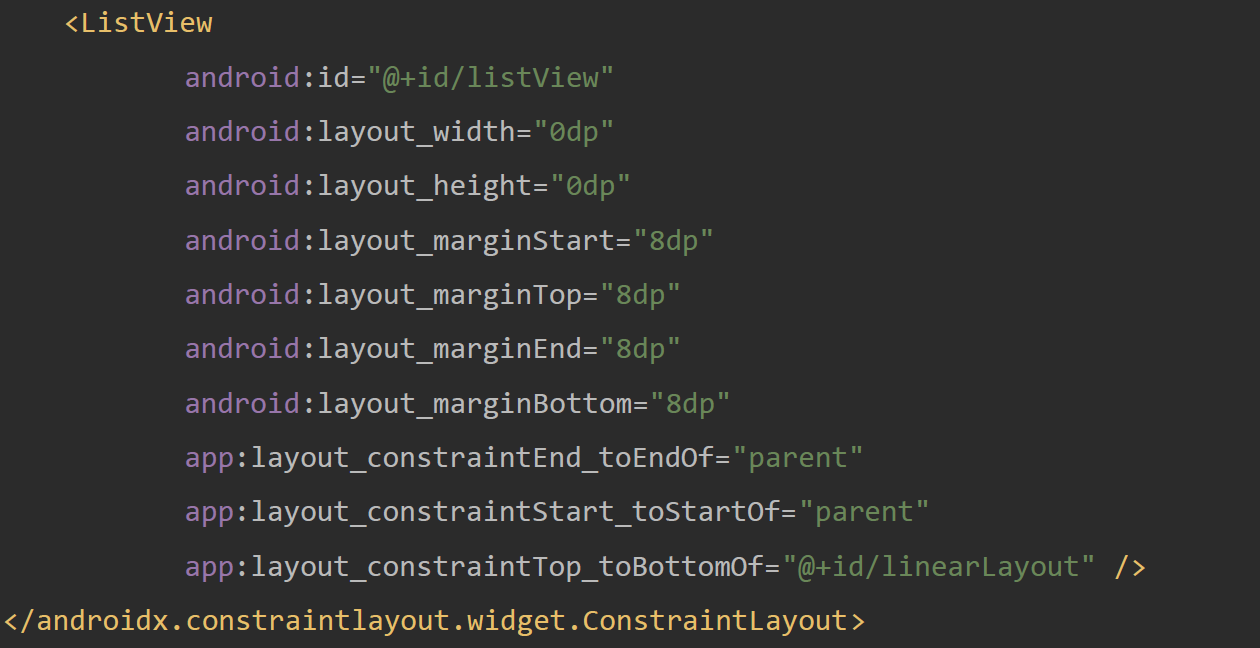
Step2 繪製activity\_main.xml檔，如圖 17所示

|  |  |
| --- | --- |
| 圖 17-1、預覽畫面 | 圖 17-2、元件樹 |

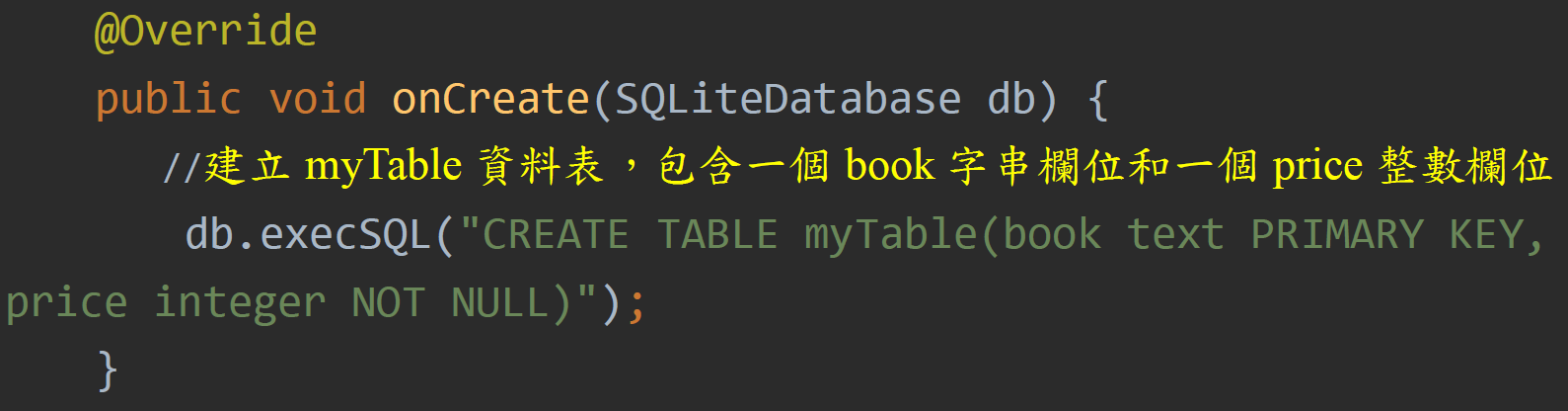
對應的xml如下：

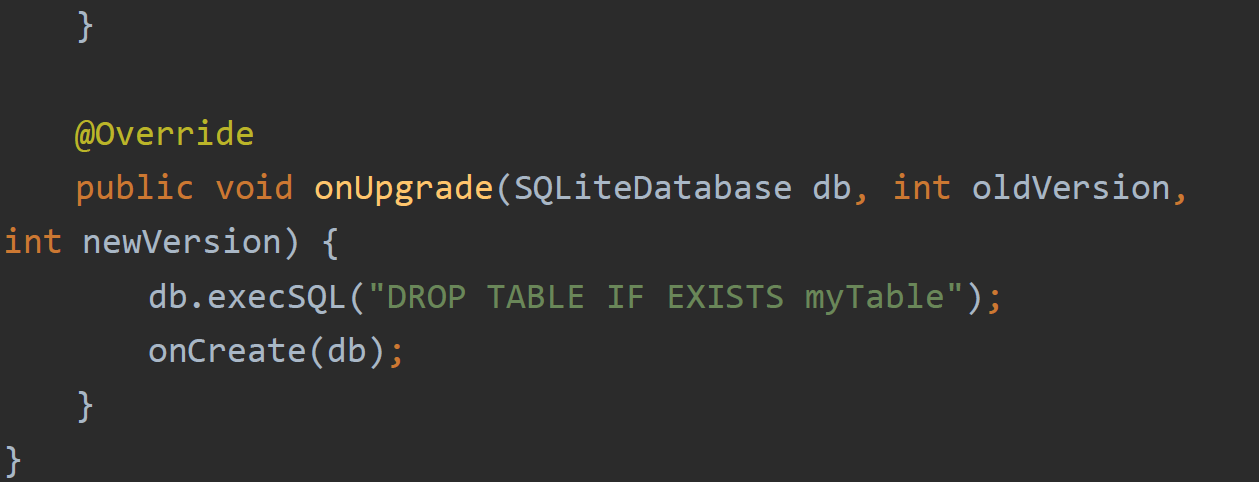
  
  
  

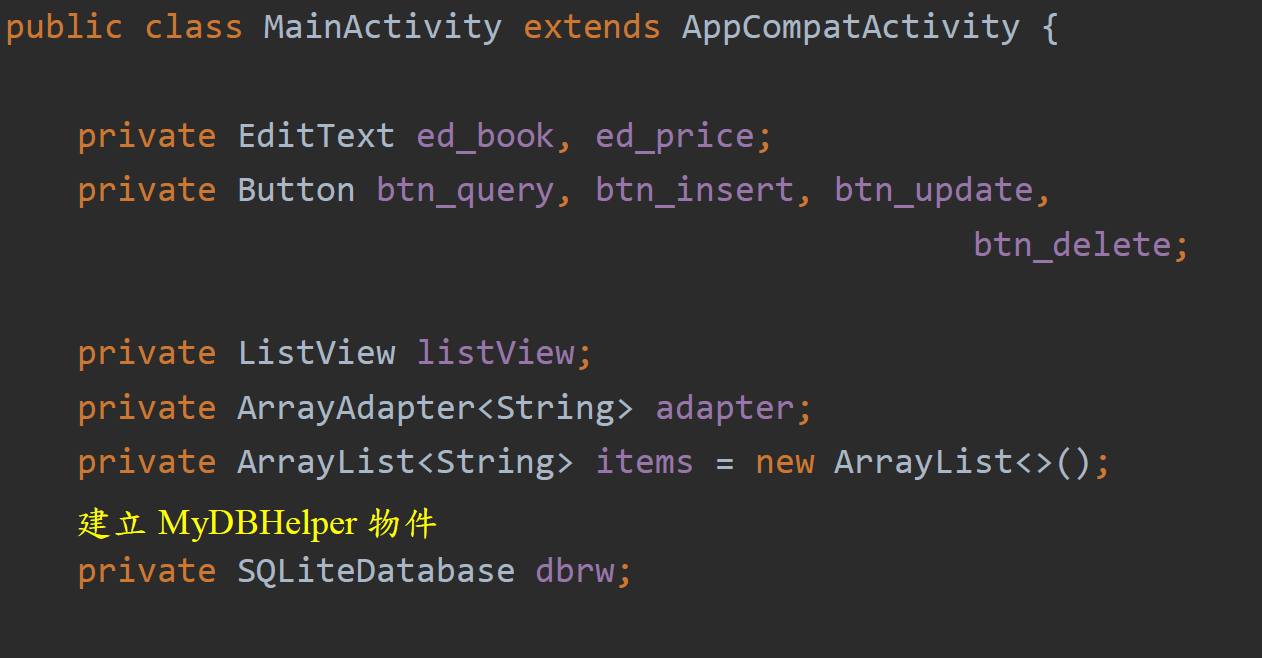
  
  
  
  


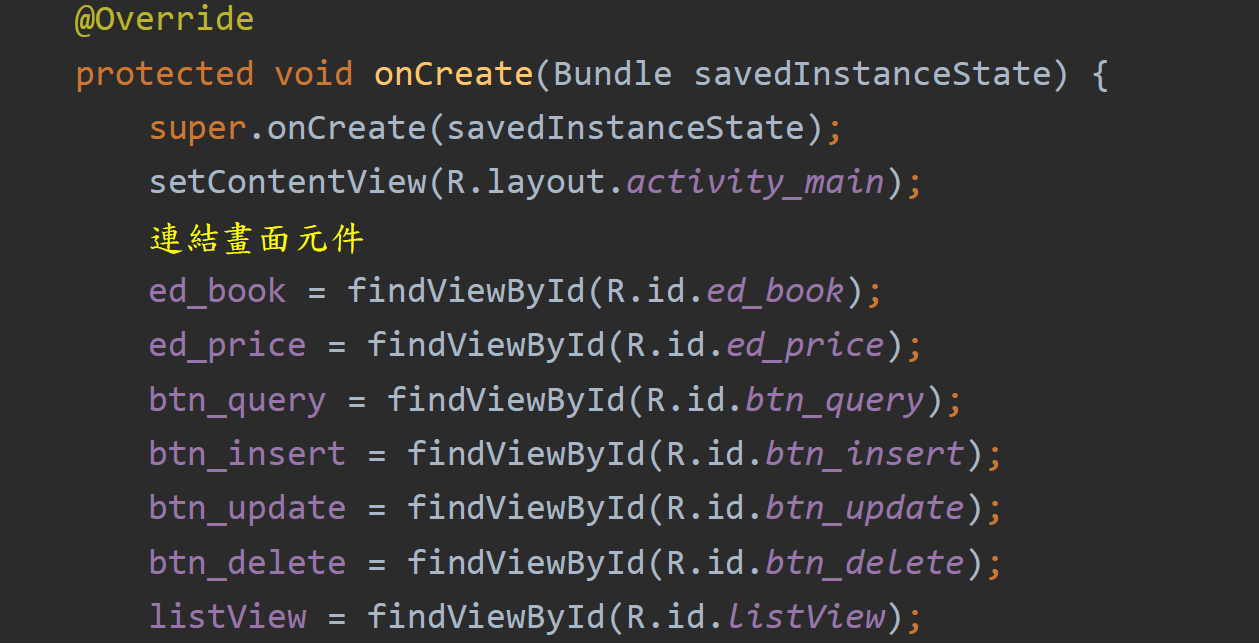
Step3 撰寫MyDBHelper，需要建立myTable資料表，包含book字串欄位、一個price整數欄位。



Step4 撰寫MainActivity，建立MyDBHelper實體，並透過getWritableDatabase()來取得SQLiteDatabase實體







Step5 為4個按鈕分別建立監聽事件，並在按下時執行對應的副程式。



