SQLITE 활용하기

기초 원리 학습

OneKick

- SQLite
  - 관계형 데이터 베이스 관리 시스템(Relational Database Management System)
  - 2000년 리처드 힙 박사 개발한 작지만 강력한 데이터베이스 엔진
  - 사용되고 있는 플랫폼
    - Android, iOS, 심비안, FireFox 웹브라우저, Adobe AIR, skype, PHP, MAC OSX, 솔라리스 등
  - 장점
    - 무료 라이선스
    - 작은 크기(150kbyte)
    - 설치와 관리 필요 없음

기초 원리 학습

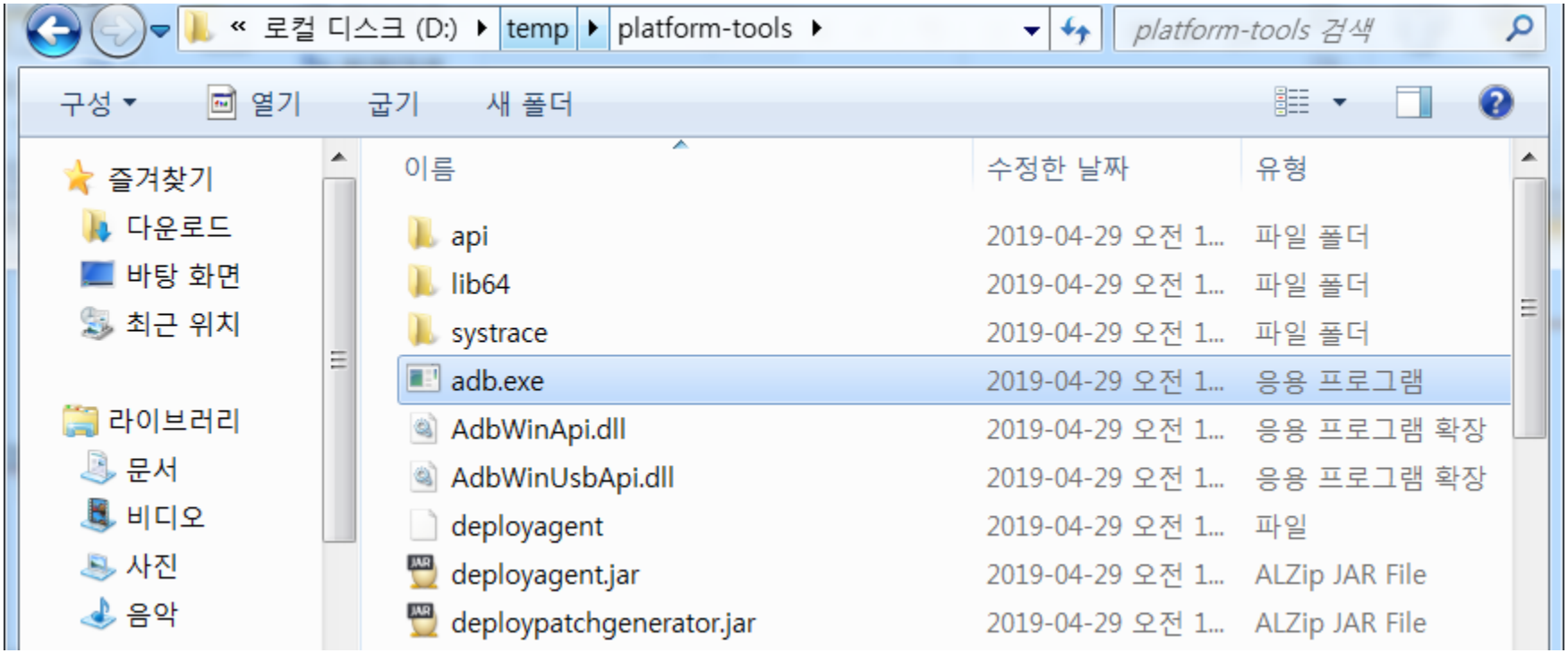
OneKick

- 단일 파일 저장 구조
  - 대부분의 임베디드 데이터베이스가 단일 파일 구조
  - DB 파일을 가져올 수 있음
    - /data/ 경로아래 설치된 앱 패키지의 /databases를 접근

기초 원리 학습

- 데이터베이스 저장 내용 확인 방법
  - adb.exe를 이용하면 됨

- 윈도우에서 command prompt 를 열기
  - android SDK 의 tools 폴더 밑으로 이동
  - adb devices 실행
  - 장치 목록 확인 가능



Adb

OneKick

- adb 사용 예시
  - C: ₩andDev ₩android-sdk-windows ₩tools>adb devices
    - 가용한 디바이스 리스트 장비에 접근하기
  - C: ₩andDev ₩android-sdk-windows ₩tools>adb-s emulator-5554 shell
    - # 로 바뀌면서 adb shell 로 들어감
  - # cd /data/data/com.android.examples.DBTester1/databases/
    - 데이터베이스 파일이 있는 곳으로 이동
  - sqlite3.exe 를 데이터베이스 이름과 같이 실행
    - #qulite3 DB명
    - (참고 : http://www.sqlite.org/sqlite.html)

기초 원리 학습

OneKick

- SQL 구문 구성 요소
  - Data Definition Language (DDL)
    - SQL 문법
  - 수정 (Modification)
    - 데이터 쓰기
  - 쿼리 (Query)
    - 데이터 읽기

기초 원리 학습

OneKick

- SQLite를 이용하여 프로젝트를 구현할 때 주의해야 할 점
  - SQLite의 열 타입은 엄격하지 않고 개발자를 위한 힌트에 불과
    - 타입 체크를 하지 않음
  - int 열에 Character를 저장하거나 또는 Character 배열에 int 형을 넣어도 에러가 발생하지 않음
  - 빠른 성능 대신 개발자의 책임 증가

기초 원리 학습

OneKick

- SQLite
  - Data Definition Language (DDL)를 기반으로 테이블과 열의 명칭들을 정의
  - 데이터베이스는 파일로 존재하고, 파일은 여러 개의 테이블로 구성
  - 테이블은 여러 행들로 구성되고, 다시 행은 여러 열 들로 구성
  - 각 열 들은 이름과 데이터 타입을 가짐

데이터베이스 > 파일 > 테이블 > 행 > 열

- 테이블 생성
  - CREATE TABLE 테이블 명 (열이름 열타입,열이름 열타입,열이름 열타입, ...);
  - 예제 : 열이 세 개인 테이블 생성
    - CREATE TABLE tableName(_id INTEGER PRIMARY KEY AUTOINCREMENT, product TEXT, price INT);

기초 원리 학습

- 수정(Modification) 방법
  - 레코드 삽입, 삭제, 업데이트하는 구문.
  - INSERT INTO TABLE tableName VALUES (null, '홍길동', '123-4567');

기초 원리 학습

OneKick

- 쿼리(Query)문 사용법 : 테이블의 데이터는 SELECT 구문으로 접근
  - SELECT * FROM tableName WHERE (_id=3);
  - SELECT name, phone FROM tableName WHERE (name LIKE “%mi%”);

기초 원리 학습

OneKick

- SQLiteOpenHelper Class
  - SQLite 기본 클래스
  - DB를 생성
  - 버전관리

- SQLiteDatabase Class
  - 쿼리부터 db 관리
  - insert() : 추가
  - delete() : 삭제
  - query() : 읽기
  - execSQL() : SQL문 직접 실행

기초 원리 학습

OneKick

- Cursor 인터페이스
  - 데이터베이스에 요청한 데이터 조회 결과를 임의로 접근 할 수 있도록 함
  - select 결과가 너무 클 수 있으므로 운영체제에게 맡기고 첫 번째 데이터의 주소만 가져옴

- ContentValues
  - insert 수행 시 여러 필드의 값을 한번에 전달 하기 위한 클래스
  - Table에 새 레코드 추가 시 사용

기초 원리 학습

OneKick

- SQLite 예제 구현 순서
  - SQLiteOpenHelper Class 구현
    - onCreate(..), onUpgrade(...) 오버라이딩 해야 함
  - SQLiteOpenHelper 클래스에서 SQLiteDatabase 객체 얻어 오기
    - Insert 시에는 getWritableDatabase()로 얻어와야 한다.
  - SQLiteOpenHelper 클래스의 onCreate()
    - 처음 한번 호출 되며 이때 테이블을 생성한다.
    - 이미 생성되어 있으면 무시된다.
  - Select 실행 후 Cursor Interface를 이용해서 결과를 반복적으로 처리

기초 원리 학습

OneKick

- SQLiteOpenHelper 클래스
  - 필수적으로 오버라이드 해야하는 메소드
    - onCreate(SQLiteDatabase) : DB 생성 시 호출
    - onUpgrade(SQLiteDatabase, int, int) : DBMS가 업그레이드 되었을 때 호출

- SQLiteDatabaseClass의 계층 구조

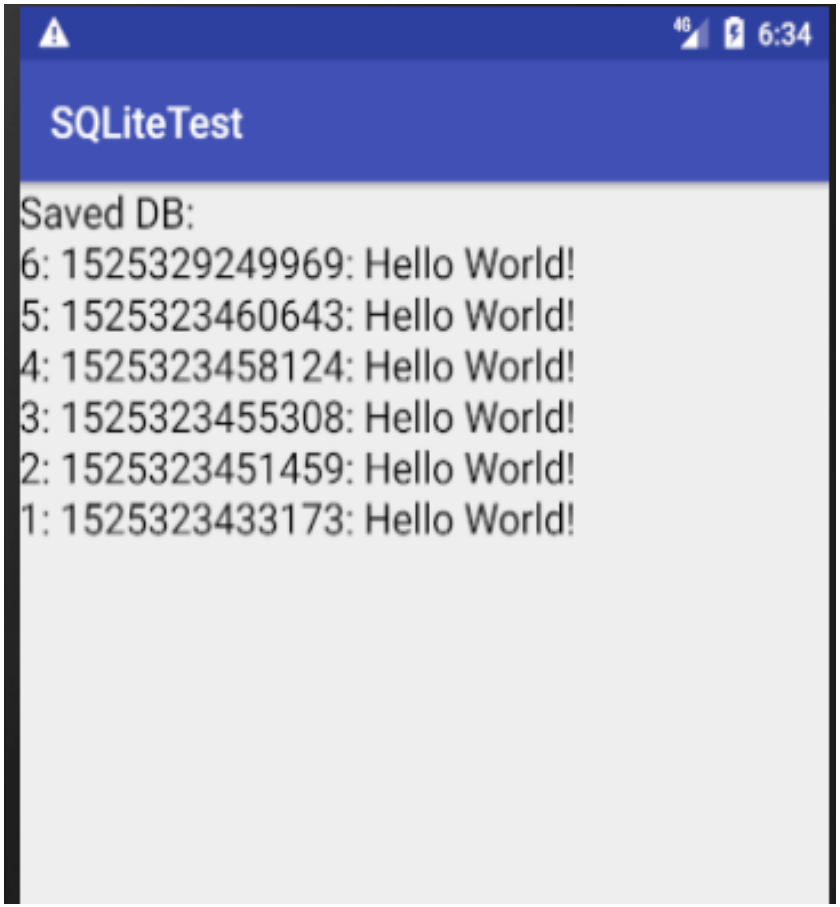
java.lang.Object

↳android.database.sqlite.SQLiteClosable

↳android.database.sqlite.SQLiteDatabase

구현 목표

- 실행할 때마다 데이터베이스의 내용이 추가되는 예제



activity_main.xml 파일

OneKick

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="안드로이드 APK 파일의 스키마 경로는 생략합니다."
    android:layout_width="match_parent" // 부모를 안벗어나게 최대화
    android:layout_height="match_parent" // 부모를 안벗어나게 최대화
    >

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent" // 부모를 안벗어나게 최대화
        android:layout_height="wrap_content" // 컨텐츠에 맞게 최소화
        android:text="Hello World!"
        android:textSize="20dp"
        android:textColor="#000000"
        />
</ScrollView>
```

MainActivity.java -1/3

OneKick

```
package com.iot.sqlitetest;

import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.support.v7.app.AppCompatActivity; // 액티비티 부모 클래스
import android.os.Bundle; // 액티비티 생성 번들
import android.widget.TextView;
import static com.iot.sqlitetest.DBOpenHelper.TABLE_NAME;
import static com.iot.sqlitetest.DBOpenHelper.TIME;
import static com.iot.sqlitetest.DBOpenHelper.TITLE;
import static com.iot.sqlitetest.DBOpenHelper._ID;

public class MainActivity extends AppCompatActivity { // 메인 화면
        private static final String DB_NAME = "MyDB";
        private static final int DB_VERSION = 1;
        private DBOpenHelper openHelper;
```

MainActivity.java -2/3

OneKick

```
@Override // 부모 메소드 재정의
protected void onCreate(Bundle savedInstanceState) { // 화면생성 이벤트
    super.onCreate(savedInstanceState); // 부모 생성자 호출
    setContentView(R.layout.activity_main); // 메인 화면 표시
    openHelper = new DBOpenHelper(this,DB_NAME,null, DB_VERSION);
    writeDB("Hello World!");
    Cursor cursor = readDB();
    displayDB(cursor);
    openHelper.close();
}

private void displayDB(Cursor cursor) {
    StringBuilder builder = new StringBuilder("Saved DB:₩n");
    while (cursor.moveToNext()) {
        long id = cursor.getLong(0);
        long time = cursor.getLong(1);
        String title = cursor.getString(2);
        builder.append(id).append(": ");
        builder.append(time).append(": ");
        builder.append(title).append("₩n");
    }
```

MainActivity.java -3/3

OneKick

```
        TextView textView = (TextView) findViewById(R.id.textView);
        textView.setText(builder);
    }

    private Cursor readDB() {
        SQLiteDatabase db = openHelper.getReadableDatabase();
        String[] from = { _ID, TIME, TITLE, };
        Cursor cursor= db.query(TABLE_NAME, from, null,
        null, null,null, TIME + " " + "DESC");
        startManagingCursor(cursor);
        return cursor;
    }

    private void writeDB(String title) {
        SQLiteDatabase db = openHelper.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(TIME, System.currentTimeMillis());
        values.put(TITLE, title);
        db.insertOrThrow(TABLE_NAME, null, values);
    }
}
```

DBOpenHelper.java -1/2

OneKick

```
package com.iot.sqlitetest;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DBOpenHelper extends SQLiteOpenHelper {

    public static final String TABLE_NAME = "MyTable";
    public static final String _ID = "_id";
    public static final String TIME = "time";
    public static final String TITLE = "title";

    public DBOpenHelper(Context context, String name,
        SQLiteDatabase.CursorFactory factory, int version) {
        super(context, name, factory, version);
    }
```

DBOpenHelper.java -2/2

OneKick

```
    @Override // 부모 메소드 재정의
    public void onCreate(SQLiteDatabase db) {
        db.execSQL( "CREATE TABLE " + TABLE_NAME + " ("
        + _ID + " INTEGER PRIMARY KEY AUTOINCREMENT, "
        + TIME + " INTEGER,"
        + TITLE + " TEXT NOT NULL);" );
    }

    @Override // 부모 메소드 재정의
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        onCreate(db);
    }
}
```

동작 테스트

- 매번 새로 시작할 때마다 SQLite에 시간을 기록
  - 지금까지 기록한 모든 데이터를 읽어와 화면에 표시
  - 재 실행할때마다 점점 데이터가 늘어가는 것을 볼 수 있음