Aplikasi CRUD Database SQLite

Sistem operasi Android mempunyai library database sendiri yang bernama SQLite. SQLite adalah library database cross platform berukuran kecil. Sehingga sangat cocok untuk diimplementasikan pada perangkat mobile yang mempunyai kapasitas ruang penyimpanan terbatas.

Membuat Aplikasi Inventaris Barang

Kali ini akan dibuat aplikasi database android dalam berupa aplikasi inventaris barang sederhana, yang berfungsi untuk mencatat barang apa saja yang telah disimpan. Aplikasi inventaris yang kita buat akan menerapkan fungsi standar operasi pada database, yaitu CRUD (Create, Read, Update, Delete).

CRUD berarti aplikasi ini akan bisa melakukan fungsi create barang baru, read data barang, update/edit info barang yang sudah tersimpan, dan delete barang di database. Dan sebagaimana aplikasi-aplikasi data-driven lainnya, aplikasi ini akan menggunakan paradigma model-view-controller (MVC).

Model View Controller

Model view controller adalah suatu arsitektur perangkat lunak yang memisahkan representasi informasi dengan interaksi pengguna. Model terdiri dari data aplikasi dan fungsi-fungsi yang berhubungan dengan database, sedangkan Controller berfungsi untuk penengah, yang akan mengkonversi input sebagai perintah untuk model atau view. View sendiri berfungsi untuk menampilkan data yang diambil dari model dan diolah lewat kontroller dalam bentuk tabel, grafik, atau informasi sederhana.

Pada aplikasi inventoris ini yang menjadi objek adalah barang, barang tersebut akan mempunyai atribut berupa :

- Id
- nama_barang
- merk_barang, dan
- harga_barang

Karena itu kita nantinya akan membuat suatu objek barang yang mempunyai atribut-atribut seperti di atas. Kelas inilah yang akan berfungsi sebagai Model.

CRUD SQLite Android (I)

Pertama-tama kita buat sebuah project Android baru terlebih dahulu. Nama package nya terserah, tapi pada program ini nama package-nya adalah id.twooh.appinventory. Silahkan nanti disesuaikan. Kemudian, karena aplikasi ini menggunakan paradigma MVC. Maka kita akan buat kelas modelnya, yaitu kelas ***Barang.java***.

Kelas barang berfungsi untuk mendefinisikan objek barang beserta fungsi-fungsi dan atribut-atributnya. Seperti barang pada umumnya, mereka mempunyai nama, ID, merk, dan sebagainya.

Kode untuk kelas ***Barang.java***.

```
package id.twooh.appinventory;

public class Barang {

    private long id;
    private String nama_barang;
    private String merk_barang;
    private String harga_barang;

    public Barang()
    {

    }

    /**
     * @return the id
     */
    public long getId() {
        return id;
    }

    /**
     * @param id the id to set
     */
    public void setId(long id) {
        this.id = id;
    }

    /**
     * @return the nama_barang
     */
    public String getNama_barang() {
        return nama_barang;
    }

    /**
     * @param nama_barang the nama_barang to set
     */
    public void setNama_barang(String nama_barang) {
        this.nama_barang = nama_barang;
    }

    /**
     * @return the merk_barang
     */
```

```
    public String getMerk_barang() {
        return merk_barang;
    }

    /**
     * @param merk_barang the merk_barang to set
     */
    public void setMerk_barang(String merk_barang) {
        this.merk_barang = merk_barang;
    }

    /**
     * @return the harga_barang
     */
    public String getHarga_barang() {
        return harga_barang;
    }

    /**
     * @param harga_barang the harga_barang to set
     */
    public void setHarga_barang(String harga_barang) {
        this.harga_barang = harga_barang;
    }

    @Override
    public String toString()
    {
        return "Barang "+ nama_barang +" "+ merk_barang + " "+
harga_barang;
    }
}
```

Model nya sekarang sudah ada. Kelas tersebut hanya berisi atribut dan fungsi-fungsi getter dan setter. Setelah kelas barang selesai dibuat, kita akan melanjutkan dengan membuat kelas Database Helper.

Database Helper

Sesuai namanya, kelas ini berfungsi untuk memudahkan kita dalam membuat dan mengakses database SQLite yang akan dipakai oleh aplikasi. Kelas ini kebanyakan berisi method-method berkaitan dengan manajemen database, seperti onUpgrade() dan onCreate().

Database Helper sebenarnya sudah ada kelasnya pada Android, yaitu bernama SQLiteOpenHelper. Sehingga kita tinggal meng-extends kelas ini dari kelas SQLiteOpenHelper.

Kode untuk kelas ***DBHelper.java*** :

```
package id.twooh.appinventory;

//deklarasi import package
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

public class DBHelper extends SQLiteOpenHelper{
        /** deklarasi konstanta-konstanta yang digunakan pada database,
seperti nama tabel,
```

```
        nama-nama kolom, nama database, dan versi dari database **/
    public static final String TABLE_NAME = "data_inventori";
    public static final String COLUMN_ID = "_id";
    public static final String COLUMN_NAME = "nama_barang";
    public static final String COLUMN_MERK = "merk_barang";
    public static final String COLUMN_HARGA = "harga_barang";
    private static final String db_name ="inventori.db";
    private static final int db_version=1;

    // Perintah SQL untuk membuat tabel database baru
      private static final String db_create = "create table "
          + TABLE_NAME + "("
          + COLUMN_ID +" integer primary key autoincrement, "
          + COLUMN_NAME+ " varchar(50) not null, "
          + COLUMN_MERK+ " varchar(50) not null, "
          + COLUMN_HARGA+ " varchar(50) not null);";

    public DBHelper(Context context) {
        super(context, db_name, null, db_version);
        // Auto generated
    }

        //mengeksekusi perintah SQL di atas untuk membuat tabel database
baru
    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(db_create);
    }

        // dijalankan apabila ingin mengupgrade database
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {
        Log.w(DBHelper.class.getName(),"Upgrading database from version " +
oldVersion + " to "
                + newVersion + ", which will destroy all old data");
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        onCreate(db);

    }

}
```

Kelas DBHelper.java kebanyakan berisi konstanta-konstanta seperti nama tabel, nama database, nama kolom, dan sebagainya. Hal ini untuk memudahkan kita ketika ingin menggunakan konstanta tersebut, tidak perlu lagi menulis nama tabel berulang kali, namun cukup dengan mengaksesnya dari kelas **DBHelper**.

Kemudian ada juga konstanta berupa SQL statement, yang untuk mereka yang ahli database, sebenarnya perintah itu digunakan untuk membuat tabel database baru. Perintah SQL tersebut kemudian akan dieksekusi pada method onCreate().

CRUD SQLite Android (II) : Membuat Fungsi Create Data

Pada bagian kedua tentang tutorial aplikasi inventaris Android menggunakan database SQLite ini, kita akan belajar tentang bagaimana membuat fungsi create/insert data, untuk memasukkan data baru ke dalam database SQLite.

Buka kembali project Aplikasi Inventaris kalian. Di pertemuan sebelumnya kita sudah membuat kelas Barang.java yang berfungsi sebagai Model objek barang, dan kelas ***DBHelper.java*** yang berfungsi untuk mempermudah aplikasi dalam membuat dan mengakses database. Selanjutnya kita akan membuat sebuah kelas yang berfungsi untuk *insert data* atau *create data*. Kelas ini berfungsi untuk menambahkan data barang baru ke database. Dan inilah daftar file-file yang akan kita buat :

- **menu.xml** : berfungsi untuk membuat sebuah layout main menu
- **create_data.xml** : berfungsi sebagail layout untuk tampilan input data
- **DBDataSource.java** : berfungsi sebagai Controller, kelas inilah yang nantinya berguna untuk melakukan operasi-operasi pada database, termasuk operasi Create Data
- **CreateData.java** : berfungsi sebagai View atau interface untuk memasukkan data

Pertama-tama, kita akan membuat kelas Menu terlebih dahulu, karena itu kita butuh file layout-nya. Buka folder res/layout pada proyek kalian, dan buat sebuah file xml baru dengan nama ***menu.xml*** Isikan kode berikut :

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    >

    <TextView
        android:id="@+id/nama_app"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:textSize="20sp"
        android:text="twoh.co"
        />

    <Button
        android:id="@+id/button_tambah"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/nama_app"
        android:layout_centerHorizontal="true"
        android:text="Tambah Data"
        />
</RelativeLayout>
```

Setelah itu, kita akan membuat kelas Main Menunya, buat sebuah activity class bernama **Menu.java**

```
package id.twooh.appinventory;
```

```
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;

public class Menu extends Activity implements OnClickListener{

    private Button bTambah;

    @Override
    public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.menu);

          bTambah = (Button) findViewById(R.id.button_tambah);
          bTambah.setOnClickListener(this);

    }
    @Override
    public void onClick(View v) {
        // TODO Auto-generated method stub
        switch(v.getId())
        {
            case R.id.button_tambah :
                Intent i = new Intent(this, CreateData.class);
                startActivity(i);
                break;

        }
    }
}
```

Kelas ini berfungsi untuk menampilkan tombol “Tambah Data”, yang apabila kita klik akan berpindah ke Activity Create Data. Karena kelas Create Data nya belum ada, selanjutnya kita akan membuat kelas tersebut. Namun kita akan membuat file xml layout-nya terlebih dahulu, yang bernama ***create_data.xml*** :

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    >

    <EditText
        android:id="@+id/nama_barang"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="15dp"
        android:inputType="text"
        android:hint=Nama Barang"
        android:ems="10" >

        <requestFocus />
    </EditText>
```

```
        <EditText
        android:id="@+id/merk_barang"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="15dp"
        android:inputType="text"
        android:hint="Merk Barang"
        android:ems="10" >

        <requestFocus />
    </EditText>
        <EditText
        android:id="@+id/harga_barang"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="15dp"
        android:inputType="number"
        android:hint="Harga Barang"
        android:ems="10" >

        <requestFocus />
    </EditText>

    <Button
        android:id="@+id/buttom_submit"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Submit"
        />
</LinearLayout>
```

Layout di atas berisi *EditText* yang berfungsi sebagai wadah inputan tiga buah informasi dasar, yaitu **nama barang**, **merk barang** dan **harga barang**, dan sebuah tombol submit di bawahnya. Setelah itu, kita akan buat Activity ***CreateData***. Aktivitas ini berfungsi sebagai View dalam paradigma Model-View-Controller yang kita gunakan. Atau dengan kata lain kelas ini menyediakan sebuah interface untuk menambahkan barang. Langsung saja kita buat kelas baru bernama ***CreateData.java***

```
package id.twooh.appinventory;

import android.os.Bundle;
import android.app.Activity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class CreateData extends Activity implements OnClickListener{

    //inisilisasi elemen-elemen pada layout
    private Button buttonSubmit;
    private EditText edNama;
    private EditText edMerk;
    private EditText edHarga;
    //inisialisasi kontroller/Data Source
    private DBDataSource dataSource;

    @Override
```

```
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.create_data);

        buttonSubmit = (Button) findViewById(R.id.buttom_submit);
        buttonSubmit.setOnClickListener(this);
        edNama = (EditText) findViewById(R.id.nama_barang);
        edHarga = (EditText) findViewById(R.id.harga_barang);
        edMerk = (EditText) findViewById(R.id.merk_barang);

        // instanstiasi kelas DBDataSource
        dataSource = new DBDataSource(this);

        //membuat sambungan baru ke database
        dataSource.open();
    }

    //KETIKA Tombol Submit Diklik
    @Override
    public void onClick(View v) {
        // Inisialisasi data barang
        String nama = null;
        String merk = null;
        String harga = null;
        @SuppressWarnings("unused")

        //inisialisasi barang baru (masih kosong)
        Barang barang = null;
        if(edNama.getText()!=null && edMerk.getText()!=null &&
edHarga.getText()!=null)
        {
            /* jika field nama, merk, dan harga tidak kosong
             * maka masukkan ke dalam data barang*/
            nama = edNama.getText().toString();
            merk = edMerk.getText().toString();
            harga = edHarga.getText().toString();
        }

        switch(v.getId())
        {
            case R.id.buttom_submit:
                // insert data barang baru
                barang = dataSource.createBarang(nama, merk, harga);

                //konfirmasi kesuksesan
                Toast.makeText(this, "masuk Barang\n" +
                        "nama" + barang.getNama_barang() +
                        "merk" + barang.getMerk_barang() +
                        "harga" + barang.getHarga_barang(),
Toast.LENGTH_LONG).show();
                break;
        }

    }
}
```

Sekarang kita sudah membuat sebuah interface/antarmuka untuk menginputkan data. Yang kita perlukan sekarang adalah sebuah Controller yang akan menyambungkan interface tersebut dengan

database yang kita buat. Jika kalian copy pastekan kode di atas akan ada error, tapi biarkan saja. Karena error tersebut disebabkan oleh kelas kontroler DBDataSource yang belum kita buat.

Langsung kita buat saja file baru bernama ***DBDataSource.java***. Kodenya seperti berikut :

```
package id.twooh.appinventory;

import java.util.ArrayList;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
import android.util.Log;

public class DBDataSource {

    //inisialiasi SQLite Database
    private SQLiteDatabase database;

    //inisialisasi kelas DBHelper
    private DBHelper dbHelper;

    //ambil semua nama kolom
    private String[] allColumns = { DBHelper.COLUMN_ID,
              DBHelper.COLUMN_NAME,
DBHelper.COLUMN_MERK,DBHelper.COLUMN_HARGA};

    //DBHelper diinstantiasi pada constructor
    public DBDataSource(Context context)
    {
        dbHelper = new DBHelper(context);
    }

    //membuka/membuat sambungan baru ke database
    public void open() throws SQLException {
        database = dbHelper.getWritableDatabase();
    }

    //menutup sambungan ke database
    public void close() {
        dbHelper.close();
    }

    //method untuk create/insert barang ke database
    public Barang createBarang(String nama, String merk, String harga) {

        // membuat sebuah ContentValues, yang berfungsi
        // untuk memasangkan data dengan nama-nama
        // kolom pada database
        ContentValues values = new ContentValues();
        values.put(DBHelper.COLUMN_NAME, nama);
        values.put(DBHelper.COLUMN_MERK, merk);
        values.put(DBHelper.COLUMN_HARGA, harga);

        // mengeksekusi perintah SQL insert data
        // yang akan mengembalikan sebuah insert ID
        long insertId = database.insert(DBHelper.TABLE_NAME, null,
            values);
```

```
        // setelah data dimasukkan, memanggil
        // perintah SQL Select menggunakan Cursor untuk
        // melihat apakah data tadi benar2 sudah masuk
        // dengan menyesuaikan ID = insertID
        Cursor cursor = database.query(DBHelper.TABLE_NAME,
            allColumns, DBHelper.COLUMN_ID + " = " + insertId, null,
            null, null, null);

        // pindah ke data paling pertama
        cursor.moveToFirst();

        // mengubah objek pada kursor pertama tadi
        // ke dalam objek barang
        Barang newBarang = cursorToBarang(cursor);

        // close cursor
        cursor.close();

        // mengembalikan barang baru
        return newBarang;
      }

    private Barang cursorToBarang(Cursor cursor)
    {
        // buat objek barang baru
        Barang barang = new Barang();
        // debug LOGCAT
        Log.v("info", "The getLONG "+cursor.getLong(0));
        Log.v("info", "The setLatLng
"+cursor.getString(1)+","+cursor.getString(2));

        /* Set atribut pada objek barang dengan
         * data kursor yang diambil dari database*/
        barang.setId(cursor.getLong(0));
        barang.setNama_barang(cursor.getString(1));
        barang.setMerk_barang(cursor.getString(2));
        barang.setHarga_barang(cursor.getString(3));

        //kembalikan sebagai objek barang
        return barang;
    }
}
```

Untuk penjelasan tiap baris kodenya, bisa dibaca pada komentar-komentar yang ada di atas. Adapun cara kerjanya sebagai berikut, pertama-tama kita kembali ke kelas ***CreateData.java***. Apabila kita telah mengisi data barang dan mengklik tombol submit, data barang tersebut akan dilempar ke Controller. Kemudian kontroler akan menerima inputan data barang tersebut dan memasangkannya sesuai dengan nama kolom pada database menggunakan ContentValues. Untuk selanjutnya dimasukkan ke database.

Setelah dimasukkan, data akan dicek kembali menggunakan SQL select yang mengembalikan data berupa Cursor. Data kursor tersebut kemudian akan diubah menjadi objek barang, dan dilemparkan lagi ke kelas ***CreateData*** untuk kemudian mengkonfirmasikan data apa saja yang barusan kita masukkan.

Yang terakhir, kita akan memodifikasi Android Manifest. Kita pindahkan intent filter launcher nya ke Activity Menu supaya aktivitas tersebut yang dijalankan pertama kali apabila aplikasi dibuka.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="id.twooh.appinventory"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="id.twooh.appinventory.CreateData"
            android:label="@string/title_activity_main" >
        </activity>
        <activity
            android:name=".Menu"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

--- Selamat mencoba ---

CRUD SQLite Android (III) : Membuat Fungsi View/Read Data

Kita akan belajar bagaimana membuat fungsi read/view data, untuk melihat data yang sudah dimasukkan ke database SQLite.

Membuat fitur Lihat Data

pada tutorial kali ini kita akan menambahkan fitur lihat data barang pada aplikasi inventaris yang akan kita buat. Pertama-tama kita akan memperbarui file layout ***menu.xml*** dengan menambahkan satu buah tombol Lihat Data.

Sebelumnya buka dulu file values/***strings.xml*** untuk memasukkan konstanta-konstanta String yang digunakan pada aplikasi.

File values/strings.xml

```
<resources>
    <string name="app_name">AppInventory</string>
    <string name="nama_app">twoh.co</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">AppInventory</string>
    <string name="barang_hint">Nama Barang</string>
    <string name="harga_hint">Harga Barang</string>
    <string name="merk_hint">Merk Barang</string>
    <string name="tombol_submit">Submit</string>
    <string name="tombol_tambah">Tambah Data</string>
    <string name="tombol_lihat">Lihat Data</string>
    <string name="title_view">Data Barang</string>
</resources>
```

Kemudian, inilah file menu.xml yang baru :

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    >

    <TextView
        android:id="@+id/nama_app"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:textSize="20sp"
        android:text="@string/nama_app"
        />

    <Button
        android:id="@+id/button_tambah"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:layout_below="@id/nama_app"
        android:layout_centerHorizontal="true"
        android:text="@string/tombol_tambah"
        />

    <Button
        android:id="@+id/button_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/button_tambah"
        android:layout_centerHorizontal="true"
        android:text="@string/tombol_lihat"
        />
</RelativeLayout>
```

Setelah itu, kita update juga file activity ***Menu.java***-nya.

```
package id.twooh.appinventory;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;

public class Menu extends Activity implements OnClickListener{

    private Button bTambah;
    private Button bLihat;
    @Override
    public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.menu);

          bTambah = (Button) findViewById(R.id.button_tambah);
          bTambah.setOnClickListener(this);
          bLihat = (Button) findViewById(R.id.button_view);
          bLihat.setOnClickListener(this);
    }
    @Override
    public void onClick(View v) {
        // TODO Auto-generated method stub
        switch(v.getId())
        {
            case R.id.button_tambah :
                Intent i = new Intent(this, CreateData.class);
                startActivity(i);
                break;
            case R.id.button_view :
                Intent i2 = new Intent(this, ViewData.class);
                startActivity(i2);
                break;
        }
    }
}
```

Kita menambahkan button kedua, yaitu ***button_view***, yang apabila di-klik akan membawa kita ke aktivitas ViewData yang akan kita buat. Apabila ada error, biarkan saja untuk sementara.

Sekarang kita akan meng-update kontroller aplikasi, yaitu file ***DataSource.java*** untuk menambahkan fungsi ***getAllBarang()*** yang berfungsi untuk mengambil semua data barang dari database. Seperti inilah fungsinya. Tambahkan pada bagian sebelum kurung tutup terakhir :

```
//mengambil semua data barang
public ArrayList<Barang> getAllBarang() {
    ArrayList<Barang> daftarBarang = new ArrayList<Barang>();

    // select all SQL query
    Cursor cursor = database.query(DBHelper.TABLE_NAME,
        allColumns, null, null, null, null, null);

    // pindah ke data paling pertama
    cursor.moveToFirst();
    // jika masih ada data, masukkan data barang ke
    // daftar barang
    while (!cursor.isAfterLast()) {
      Barang barang = cursorToBarang(cursor);
      daftarBarang.add(barang);
      cursor.moveToNext();
    }
    // Make sure to close the cursor
    cursor.close();
    return daftarBarang;
  }
```

Dan ini adalah kode lengkap untuk ***DBDataSource.java*** :

```
package id.twooh.appinventory;

import java.util.ArrayList;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
import android.util.Log;

public class DBDataSource {

    //inisialiasi SQLite Database
    private SQLiteDatabase database;

    //inisialisasi kelas DBHelper
    private DBHelper dbHelper;

    //ambil semua nama kolom
    private String[] allColumns = { DBHelper.COLUMN_ID,
              DBHelper.COLUMN_NAME,
DBHelper.COLUMN_MERK,DBHelper.COLUMN_HARGA};

    //DBHelper diinstantiasi pada constructor
    public DBDataSource(Context context)
    {
        dbHelper = new DBHelper(context);
```

```
    }

    //membuka/membuat sambungan baru ke database
    public void open() throws SQLException {
        database = dbHelper.getWritableDatabase();
    }

    //menutup sambungan ke database
    public void close() {
        dbHelper.close();
    }

    //method untuk create/insert barang ke database
     public Barang createBarang(String nama, String merk, String harga) {

        // membuat sebuah ContentValues, yang berfungsi
        // untuk memasangkan data dengan nama-nama
        // kolom pada database
        ContentValues values = new ContentValues();
        values.put(DBHelper.COLUMN_NAME, nama);
        values.put(DBHelper.COLUMN_MERK, merk);
        values.put(DBHelper.COLUMN_HARGA, harga);

        // mengeksekusi perintah SQL insert data
        // yang akan mengembalikan sebuah insert ID
        long insertId = database.insert(DBHelper.TABLE_NAME, null,
            values);

        // setelah data dimasukkan, memanggil
        // perintah SQL Select menggunakan Cursor untuk
        // melihat apakah data tadi benar2 sudah masuk
        // dengan menyesuaikan ID = insertID
        Cursor cursor = database.query(DBHelper.TABLE_NAME,
            allColumns, DBHelper.COLUMN_ID + " = " + insertId, null,
            null, null, null);

        // pindah ke data paling pertama
        cursor.moveToFirst();

        // mengubah objek pada kursor pertama tadi
        // ke dalam objek barang
        Barang newBarang = cursorToBarang(cursor);

        // close cursor
        cursor.close();

        // mengembalikan barang baru
        return newBarang;
      }

    private Barang cursorToBarang(Cursor cursor)
    {
        // buat objek barang baru
        Barang barang = new Barang();
        // debug LOGCAT
        Log.v("info", "The getLONG "+cursor.getLong(0));
        Log.v("info", "The setLatLng
"+cursor.getString(1)+","+cursor.getString(2));
```

```
        /* Set atribut pada objek barang dengan
         * data kursor yang diambil dari database*/
        barang.setId(cursor.getLong(0));
        barang.setNama_barang(cursor.getString(1));
        barang.setMerk_barang(cursor.getString(2));
        barang.setHarga_barang(cursor.getString(3));

        //kembalikan sebagai objek barang
        return barang;
    }

    //mengambil semua data barang
    public ArrayList<Barang> getAllBarang() {
        ArrayList<Barang> daftarBarang = new ArrayList<Barang>();

        // select all SQL query
        Cursor cursor = database.query(DBHelper.TABLE_NAME,
            allColumns, null, null, null, null, null);

        // pindah ke data paling pertama
        cursor.moveToFirst();
        // jika masih ada data, masukkan data barang ke
        // daftar barang
        while (!cursor.isAfterLast()) {
          Barang barang = cursorToBarang(cursor);
          daftarBarang.add(barang);
          cursor.moveToNext();
        }
        // Make sure to close the cursor
        cursor.close();
        return daftarBarang;
      }
}
```

Kontroller sudah diperbarui, sekarang tinggal membuat view baru untuk menampilkan data barang yang ada pada database. Kita akan membuat layoutnya dulu, buat file xml layout baru dengan nama ***viewdata.xml***. Dan isikan kode berikut :

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    >

    <TextView
        android:id="@+id/data_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/title_view"
        android:layout_gravity="center_horizontal"
        android:background="#0000ff"
        />
    <ListView
        android:id="@android:id/list"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```

```
        />
</LinearLayout>
```

Kita menggunakan ListView layout untuk menampilkan data barang pada aplikasi. Setelah itu, kita akan membuat file activity-nya bernama ***ViewData.java***. Pada kelas tersebut, data barang yang diambil akan dimasukkan ke dalam *ArrayList* yang nantinya akan diset menggunakan *ArrayAdapter*. Ini adalah kode untuk kelas ***ViewData.java*** :

```
package id.twooh.appinventory;

import java.util.ArrayList;

import android.app.ListActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;

public class ViewData extends ListActivity {

    //inisialisasi kontroller
    private DBDataSource dataSource;

    //inisialisasi arraylist
    private ArrayList<Barang> values;
    @Override
    public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.viewdata);

          dataSource = new DBDataSource(this);
          // buka kontroller
          dataSource.open();

          // ambil semua data barang
          values = dataSource.getAllBarang();

          // masukkan data barang ke array adapter
          ArrayAdapter<Barang> adapter = new ArrayAdapter<Barang>(this,
                    android.R.layout.simple_list_item_1, values);

          // set adapter pada list
          setListAdapter(adapter);
    }
}
```

Yang terakhir adalah memodifikasi Android Manifest dengan menambahkan View Data ke dalam daftar aktivitas. Buka file AndroidManifest dan masukkan kode berikut :

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="id.twooh.appinventory"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15" />
```

```
    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="id.twooh.appinventory.CreateData"
            android:label="@string/title_activity_main" >
        </activity>
        <activity
            android:name=".Menu"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
        </activity>
        <activity
            android:name=".ViewData"
            android:label="@string/title_activity_main" >
        </activity>
    </application>

</manifest>
```

--- Selamat Mencoba ---

SQLite Database CRUD Tutorial (IV) : Membuat Fungsi Update Data

Kita akan melanjutkan belajar tentang bagaimana membuat fungsi edit/update data untuk mengedit data yang sudah dimasukkan ke dalam database SQLite.

Membuat Fitur Edit/Update Data pada SQLite Android

Fitur Update/Edit data ini dijalankan dengan pertama-tama melakukan long click item pada list view data. Yang kemudian akan memunculkan dialog dan kita bisa memilih untuk Edit data. Pertama-tama kita harus menambahkan dua buah method baru pada kelas controller yaitu kelas ***DBDataSource.java***. Yaitu method untuk select data, dan yang kedua adalah method untuk update data.

Tambahkan method ***getBarang()*** berikut ini pada ***DBDataSource.java*** :

```
//ambil satu barang sesuai id
public Barang getBarang(long id)
  {
      Barang barang = new Barang(); //inisialisasi barang
      //select query
      Cursor cursor = database.query(DBHelper.TABLE_NAME, allColumns, "_id
="+id, null, null, null, null);
      //ambil data yang pertama
      cursor.moveToFirst();
      //masukkan data cursor ke objek barang
      barang = cursorToBarang(cursor);
      //tutup sambungan
      cursor.close();
      //return barang
      return barang;
  }
```

Method itu akan mengambil satu barang berdasarkan id barang yang kita definisikan. Dan data yang sudah diambil itulah yang nantinya akan diedit.

Kemudian ini adalah method untuk update data, tambahkan juga pada ***DBDataSource.java***

```
//update barang yang diedit
    public void updateBarang(Barang b)
    {
        //ambil id barang
        String strFilter = "_id=" + b.getId();
        //memasukkan ke content values
        ContentValues args = new ContentValues();
        //masukkan data sesuai dengan kolom pada database
        args.put(DBHelper.COLUMN_NAME, b.getNama_barang());
        args.put(DBHelper.COLUMN_MERK, b.getMerk_barang());
        args.put(DBHelper.COLUMN_HARGA, b.getHarga_barang() );
        //update query
        database.update(DBHelper.TABLE_NAME, args, strFilter, null);
    }
```

Dan inilah kelas lengkap ***DBSource.java*** setelah modifikasi :

```
package id.twooh.appinventory;

import java.util.ArrayList;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;

public class DBDataSource {

    //inisialiasi SQLite Database
    private SQLiteDatabase database;

    //inisialisasi kelas DBHelper
    private DBHelper dbHelper;

    //ambil semua nama kolom
    private String[] allColumns = { DBHelper.COLUMN_ID,
              DBHelper.COLUMN_NAME,
DBHelper.COLUMN_MERK,DBHelper.COLUMN_HARGA};

    //DBHelper diinstantiasi pada constructor
    public DBDataSource(Context context)
    {
        dbHelper = new DBHelper(context);
    }

    //membuka/membuat sambungan baru ke database
    public void open() throws SQLException {
        database = dbHelper.getWritableDatabase();
    }

    //menutup sambungan ke database
    public void close() {
        dbHelper.close();
    }

    //method untuk create/insert barang ke database
    public Barang createBarang(String nama, String merk, String harga) {

        // membuat sebuah ContentValues, yang berfungsi
        // untuk memasangkan data dengan nama-nama
        // kolom pada database
        ContentValues values = new ContentValues();
        values.put(DBHelper.COLUMN_NAME, nama);
        values.put(DBHelper.COLUMN_MERK, merk);
        values.put(DBHelper.COLUMN_HARGA, harga);

        // mengeksekusi perintah SQL insert data
        // yang akan mengembalikan sebuah insert ID
        long insertId = database.insert(DBHelper.TABLE_NAME, null,
            values);

        // setelah data dimasukkan, memanggil
        // perintah SQL Select menggunakan Cursor untuk
        // melihat apakah data tadi benar2 sudah masuk
        // dengan menyesuaikan ID = insertID
        Cursor cursor = database.query(DBHelper.TABLE_NAME,
```

```
            allColumns, DBHelper.COLUMN_ID + " = " + insertId, null,
            null, null, null);

        // pindah ke data paling pertama
        cursor.moveToFirst();

        // mengubah objek pada kursor pertama tadi
        // ke dalam objek barang
        Barang newBarang = cursorToBarang(cursor);

        //close cursor
        cursor.close();

        //mengembalikan barang baru
        return newBarang;
      }

    private Barang cursorToBarang(Cursor cursor)
    {
        // buat objek barang baru
        Barang barang = new Barang();
        // debug LOGCAT
        //Log.v("info", "The getLONG "+cursor.getLong(0));
        //Log.v("info", "The setLatLng
"+cursor.getString(1)+","+cursor.getString(2));

        // Set atribut pada objek barang dengan
        // data kursor yang diambil dari database
        barang.setId(cursor.getLong(0));
        barang.setNama_barang(cursor.getString(1));
        barang.setMerk_barang(cursor.getString(2));
        barang.setHarga_barang(cursor.getString(3));

        //kembalikan sebagai objek barang
        return barang;
    }

    //mengambil semua data barang
    public ArrayList<Barang> getAllBarang() {
        ArrayList<Barang> daftarBarang = new ArrayList<Barang>();

        // select all SQL query
        Cursor cursor = database.query(DBHelper.TABLE_NAME,
            allColumns, null, null, null, null, null);

        // pindah ke data paling pertama
        cursor.moveToFirst();
        // jika masih ada data, masukkan data barang ke
        // daftar barang
        while (!cursor.isAfterLast()) {
          Barang barang = cursorToBarang(cursor);
          daftarBarang.add(barang);
          cursor.moveToNext();
        }
        // Make sure to close the cursor
        cursor.close();
        return daftarBarang;
      }
```

```
    //ambil satu barang sesuai id
    public Barang getBarang(long id)
      {
          Barang barang = new Barang(); //inisialisasi barang
          //select query
          Cursor cursor = database.query(DBHelper.TABLE_NAME, allColumns,
"_id ="+id, null, null, null, null);
          //ambil data yang pertama
          cursor.moveToFirst();
          //masukkan data cursor ke objek barang
          barang = cursorToBarang(cursor);
          //tutup sambungan
          cursor.close();
          //return barang
          return barang;
      }

    //update barang yang diedit
    public void updateBarang(Barang b)
    {
        //ambil id barang
        String strFilter = "_id=" + b.getId();
        //memasukkan ke content values
        ContentValues args = new ContentValues();
        //masukkan data sesuai dengan kolom pada database
        args.put(DBHelper.COLUMN_NAME, b.getNama_barang());
        args.put(DBHelper.COLUMN_MERK, b.getMerk_barang());
        args.put(DBHelper.COLUMN_HARGA, b.getHarga_barang() );
        //update query
        database.update(DBHelper.TABLE_NAME, args, strFilter, null);
    }
}
```

Membuat View untuk edit data

Setelah method pada kontroller ditambah. Sekarang kita akan membuat view untuk edit data. View tersebut mempunyai layout berupa tiga buah edit text untuk menampilkan data nama, merk dan harga barang. Kemudian dua buah tombol untuk save data yang sudah di-update dan tombol cancel.

Seperti biasa pertama-tama kita harus menambahkan beberapa resource string. Masukkan kode berikut pada res/values/string.xml.

```
<resources>
    <string name="app_name">AppInventory</string>
    <string name="nama_app">twoh.co</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">AppInventory</string>
    <string name="barang_hint">Nama Barang</string>
    <string name="harga_hint">Harga Barang</string>
    <string name="merk_hint">Merk Barang</string>
    <string name="tombol_submit">Submit</string>
    <string name="tombol_tambah">Tambah Data</string>
    <string name="tombol_lihat">Lihat Data</string>
    <string name="tombol_edit">Save</string>
    <string name="tombol_cancel">Cancel</string>
    <string name="tombol_dialog_edit">Edit</string>
    <string name="tombol_dialog_delete">Delete</string>
    <string name="title_view">Data Barang</string>
```

```
    <string name="title_edit">Edit Barang | ID : </string>
</resources>
```

Setelah itu, kita buat layout file baru bernama edit_data.xml. File ini adalah layout untuk Edit View. Tambahkan kode berikut pada ***edit_data.xml***.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:ads="http://schemas.android.com/apk/lib/com.google.ads"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    >
    <TextView
        android:id="@+id/text_id_barang"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/title_edit"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="15dp"
        android:background="#0000ff"
        />
    <EditText
        android:id="@+id/editText_nama"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:layout_marginTop="15dp"
        android:background="#ffffffff"
        android:layout_below="@id/text_id_barang"
        >
        <requestFocus />
    </EditText>
    <EditText
        android:id="@+id/editText_merk"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:layout_marginTop="15dp"
        android:background="#ffffffff"
        android:layout_below="@id/editText_nama"
         />
    <EditText
        android:id="@+id/editText_harga"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="number"
        android:ems="10"
        android:layout_marginTop="15dp"
        android:background="#ffffffff"
        android:layout_below="@id/editText_merk"
        />
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:gravity="center"
        android:layout_below="@id/editText_harga"
```

```
        >
        <Button
        android:id="@+id/button_save_update"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/tombol_edit"
        />

    <Button
        android:id="@+id/button_cancel_update"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/tombol_cancel"
        />
    </LinearLayout>
</RelativeLayout>
```

Update kelas View Data

Sekarang kita akan mengupdate kelas ***ViewData.java***. Kita akan menambahkan method *OnItemLongClickListener* pada item di ListView. Sehingga apabila kita melakukan long click/klik yang panjang, akan ada dialog muncul. Sebelumnya kita akan membuat layout untuk dialog terlebih dahulu. Buat sebuah file xml baru bernama ***dialog_view.xml***.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:gravity="center"
    >

    <Button
        android:id="@+id/button_edit_data"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/tombol_dialog_edit"
        />

    <Button
        android:id="@+id/button_delete_data"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/tombol_dialog_delete"
        />
</LinearLayout>
```

Kemudian, buka file ***ViewData.xml*** dan ganti dengan kode berikut

```
package id.twooh.appinventory;

import java.util.ArrayList;
import android.app.Dialog;
import android.app.ListActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
```

```
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemLongClickListener;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;

public class ViewData extends ListActivity implements
OnItemLongClickListener {

    //inisialisasi kontroller
    private DBDataSource dataSource;

    //inisialisasi arraylist
    private ArrayList<Barang> values;
    private Button editButton;
    private Button delButton;

    @Override
    public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.viewdata);
          dataSource = new DBDataSource(this);
          // buka kontroller
          dataSource.open();

          // ambil semua data barang
          values = dataSource.getAllBarang();

          // masukkan data barang ke array adapter
          ArrayAdapter<Barang> adapter = new ArrayAdapter<Barang>(this,
          android.R.layout.simple_list_item_1, values);

          // set adapter pada list
          setListAdapter(adapter);

          // mengambil listview untuk diset onItemLongClickListener
          ListView lv = (ListView) findViewById(android.R.id.list);
          lv.setOnItemLongClickListener(this);
    }

    //apabila ada long click
    @Override
    public boolean onItemLongClick(final AdapterView<?> adapter, View v,
int pos,
            final long id) {

        //tampilkan alert dialog
        final Dialog dialog = new Dialog(this);
        dialog.setContentView(R.layout.dialog_view);
        dialog.setTitle("Pilih Aksi");
        dialog.show();
        final Barang b = (Barang) getListAdapter().getItem(pos);
        editButton = (Button) dialog.findViewById(R.id.button_edit_data);
        delButton = (Button) dialog.findViewById(R.id.button_delete_data);

        //apabila tombol edit diklik
        editButton.setOnClickListener(
                new OnClickListener()
```

```
                {
                    @Override
                    public void onClick(View v) {
                        // TODO Auto-generated method stub
                        switchToEdit(b.getId());
                        dialog.dismiss();
                    }
                }
                );
        return true;
    }
    //method untuk edit data
    public void switchToEdit(long id)
    {
        Barang b = dataSource.getBarang(id);
        Intent i = new Intent(this, EditData.class);
        Bundle bun = new Bundle();
        bun.putLong("id", b.getId());
        bun.putString("nama", b.getNama_barang());
        bun.putString("merk", b.getMerk_barang());
        bun.putString("harga", b.getHarga_barang());
        i.putExtras(bun);
        finale();
        startActivity(i);
    }
        //method yang dipanggil ketika edit data selesai
    public void finale()
    {
        ViewData.this.finish();
        dataSource.close();
    }
    @Override
    protected void onResume() {
        dataSource.open();
        super.onResume();
    }

    @Override
    protected void onPause() {
        dataSource.close();
        super.onPause();
    }

}
```

Dari kode di atas kita bisa lihat bahwa sebagian besar perubahan ada pada fungsi untuk mengeset *OnClickListener* pada item di listview. Jadi cara kerjanya, apabila list item diklik dan kemudian ditahan klik itu (long click), nanti akan muncul alert dialog yang memberikan pilihan kepada kita. Apakah ingin delete data, atau edit data. Apabila kita mengklik tombol edit data, maka kita akan dibawa ke activity untuk edit data.

Membuat Activity Class EditData.java

Karena kita sudah membuat layout xml untuk edit data, sekarang kita akan membuat activity-nya. Buat file java baru bernama ***EditData.java***.

```
package id.twooh.appinventory;
```

```
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class EditData extends Activity implements OnClickListener{

    private DBDataSource dataSource;

    private long id;
    private String harga;
    private String merk;
    private String nama;

    private EditText edNama;
    private EditText edHarga;
    private EditText edMerk;

    private TextView txId;

    private Button btnSave;
    private Button btnCancel;

    private Barang barang;

    @Override
    public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.edit_data);
              //inisialisasi variabel
          edNama = (EditText) findViewById(R.id.editText_nama);
          edHarga = (EditText) findViewById(R.id.editText_harga);
          edMerk = (EditText) findViewById(R.id.editText_merk);
          txId = (TextView) findViewById(R.id.text_id_barang);
          //buat sambungan baru ke database
          dataSource = new DBDataSource(this);
          dataSource.open();
          // ambil data barang dari extras
          Bundle bun = this.getIntent().getExtras();
          id = bun.getLong("id");
          harga = bun.getString("harga");
          merk = bun.getString("merk");
          nama = bun.getString("nama");
          //masukkan data-data barang tersebut ke field editor
          txId.append(String.valueOf(id));
          edNama.setText(nama);
          edHarga.setText(harga);
          edMerk.setText(merk);

          //set listener pada tombol
          btnSave = (Button) findViewById(R.id.button_save_update);
          btnSave.setOnClickListener(this);
          btnCancel = (Button) findViewById(R.id.button_cancel_update);
          btnCancel.setOnClickListener(this);
    }
```

```
    @Override
    public void onClick(View v) {
        // TODO Auto-generated method stub
        switch(v.getId())
        {
                        // apabila tombol save diklik (update barang)
            case R.id.button_save_update :
                barang = new Barang();
                barang.setHarga_barang(edHarga.getText().toString());
                barang.setNama_barang(edNama.getText().toString());
                barang.setMerk_barang(edMerk.getText().toString());
                barang.setId(id);
                dataSource.updateBarang(barang);
                Intent i = new Intent(this, ViewData.class);
                startActivity(i);
                EditData.this.finish();
                dataSource.close();
                break;
                        // apabila tombol cancel diklik, finish activity
            case R.id.button_cancel_update :
                finish();
                dataSource.close();
                break;
        }
    }
}
```

Activity EditData.java pada dasarnya berfungsi untuk mengambil data barang yang mau diedit, seperti nama, merk, dan harga. Kemudian data tersebut ditempatkan pada field EditText supaya bisa kita ganti. Setelah selesai, maka kita mengklik tombol Save dan kemudian activity akan memanggil method *updateBarang()* pada kontroller, yang akan melakukan update data lewat SQL Query.

Tambah activity EditData.java ke Android Manifest

Terakhir adalah menambahkan activity EditData di atas ke Android Manifest. Buka file ***AndroidManifest.xml***.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="id.twooh.appinventory"
    android:versionCode="3"
    android:versionName="1.0.0.4a" >

    <uses-permission
         android:name="android.permission.INTERNET" />
    <uses-permission
         android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="16" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme"
        android:allowBackup="true"
        >
        <activity
```

```
            android:name="id.twooh.appinventory.CreateData"
            android:label="@string/title_activity_main" >
        </activity>
        <activity
            android:name=".Menu"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
        </activity>
        <activity
            android:name=".ViewData"
            android:label="@string/title_activity_main" >
        </activity>
        <activity
            android:name=".EditData"
            android:label="@string/title_activity_main" >
        </activity>
    </application>

</manifest>
```

--- Selamat Mencoba ---

Android SQLite Database CRUD Tutorial (V) : Membuat Fungsi Delete Data

Sejauh ini, aplikasi inventaris barang sudah mempunyai fitur untuk tambah data, baca data, edit data. Kita akan membahas satu fitur yang belum kita implementasikan, yaitu fitur Delete data. Dan sesuai urutannya, Create, Read, Update, Delete.

Membuat Fitur Delete data pada SQLIte Android

Seperti biasanya, kita akan menambahkan satu method baru pada kontroller ***DBDataSource.java***. Satu method yang berfungsi untuk mendelete data tertentu pada database SQLite. Buka file DBDataSource.java, dan tambahkan kode berikut.

```
package id.twooh.appinventory;

import java.util.ArrayList;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;

public class DBDataSource {

    //inisialiasi SQLite Database
    private SQLiteDatabase database;

    //inisialisasi kelas DBHelper
    private DBHelper dbHelper;

    //ambil semua nama kolom
    private String[] allColumns = { DBHelper.COLUMN_ID,
              DBHelper.COLUMN_NAME,
DBHelper.COLUMN_MERK,DBHelper.COLUMN_HARGA};

    //DBHelper diinstantiasi pada constructor
    public DBDataSource(Context context)
    {
        dbHelper = new DBHelper(context);
    }

    //membuka/membuat sambungan baru ke database
    public void open() throws SQLException {
        database = dbHelper.getWritableDatabase();
    }

    //menutup sambungan ke database
    public void close() {
        dbHelper.close();
    }

    //method untuk create/insert barang ke database
```

```
    public Barang createBarang(String nama, String merk, String harga) {

        // membuat sebuah ContentValues, yang berfungsi
        // untuk memasangkan data dengan nama-nama
        // kolom pada database
        ContentValues values = new ContentValues();
        values.put(DBHelper.COLUMN_NAME, nama);
        values.put(DBHelper.COLUMN_MERK, merk);
        values.put(DBHelper.COLUMN_HARGA, harga);

        // mengeksekusi perintah SQL insert data
        // yang akan mengembalikan sebuah insert ID
        long insertId = database.insert(DBHelper.TABLE_NAME, null,
            values);

        // setelah data dimasukkan, memanggil
        // perintah SQL Select menggunakan Cursor untuk
        // melihat apakah data tadi benar2 sudah masuk
        // dengan menyesuaikan ID = insertID
        Cursor cursor = database.query(DBHelper.TABLE_NAME,
            allColumns, DBHelper.COLUMN_ID + " = " + insertId, null,
            null, null, null);

        // pindah ke data paling pertama
        cursor.moveToFirst();

        // mengubah objek pada kursor pertama tadi
        // ke dalam objek barang
        Barang newBarang = cursorToBarang(cursor);

        //close cursor
        cursor.close();

        //mengembalikan barang baru
        return newBarang;
      }

    private Barang cursorToBarang(Cursor cursor)
    {
        // buat objek barang baru
        Barang barang = new Barang();
        // debug LOGCAT
        //Log.v("info", "The getLONG "+cursor.getLong(0));
        //Log.v("info", "The setLatLng
"+cursor.getString(1)+","+cursor.getString(2));

        // Set atribut pada objek barang dengan
        // data kursor yang diambil dari database
        barang.setId(cursor.getLong(0));
        barang.setNama_barang(cursor.getString(1));
        barang.setMerk_barang(cursor.getString(2));
        barang.setHarga_barang(cursor.getString(3));

        //kembalikan sebagai objek barang
        return barang;
    }

    //mengambil semua data barang
    public ArrayList<Barang> getAllBarang() {
```

```
        ArrayList<Barang> daftarBarang = new ArrayList<Barang>();

        // select all SQL query
        Cursor cursor = database.query(DBHelper.TABLE_NAME,
            allColumns, null, null, null, null, null);

        // pindah ke data paling pertama
        cursor.moveToFirst();
        // jika masih ada data, masukkan data barang ke
        // daftar barang
        while (!cursor.isAfterLast()) {
          Barang barang = cursorToBarang(cursor);
          daftarBarang.add(barang);
          cursor.moveToNext();
        }
        // Make sure to close the cursor
        cursor.close();
        return daftarBarang;
      }

    //ambil satu barang sesuai id
    public Barang getBarang(long id)
      {
          Barang barang = new Barang(); //inisialisasi barang
          //select query
          Cursor cursor = database.query(DBHelper.TABLE_NAME, allColumns,
"_id ="+id, null, null, null, null);
          //ambil data yang pertama
          cursor.moveToFirst();
          //masukkan data cursor ke objek barang
          barang = cursorToBarang(cursor);
          //tutup sambungan
          cursor.close();
          //return barang
          return barang;
      }

    //update barang yang diedit
    public void updateBarang(Barang b)
    {
        //ambil id barang
        String strFilter = "_id=" + b.getId();
        //memasukkan ke content values
        ContentValues args = new ContentValues();
        //masukkan data sesuai dengan kolom pada database
        args.put(DBHelper.COLUMN_NAME, b.getNama_barang());
        args.put(DBHelper.COLUMN_MERK, b.getMerk_barang());
        args.put(DBHelper.COLUMN_HARGA, b.getHarga_barang() );
        //update query
        database.update(DBHelper.TABLE_NAME, args, strFilter, null);
    }

    // delete barang sesuai ID
    public void deleteBarang(long id)
    {
        String strFilter = "_id=" + id;
        database.delete(DBHelper.TABLE_NAME, strFilter, null);
    }
}
```

Kode yang baru ada pada bagian yang di-bold, dan pada dasarnya kode itu hanya berfungsi untuk melakukan sebuah SQL query untuk delete data di database.

Menambahkan tombol Delete pada ViewData activity

Kemudian kita akan mengupdate kelas View Data, sebenarnya kita sudah mempunyai tombol delete pada dialog yang akan dimunculkan ketika kita melakukan long click pada data item. Yang harus kita lakukan sekarang adalah mengeset method yang akan dijalankan apabila tombol delete tersebut diklik. Ganti kode pada kelas ***ViewData.java*** dengan kode di bawah ini :

```
package id.twooh.appinventory;

import java.util.ArrayList;

import android.app.Dialog;
import android.app.ListActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemLongClickListener;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;

public class ViewData extends ListActivity implements
OnItemLongClickListener {

    //inisialisasi kontroller
    private DBDataSource dataSource;

    //inisialisasi arraylist
    private ArrayList<Barang> values;
    private Button editButton;
    private Button delButton;

    @Override
    public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.viewdata);
          dataSource = new DBDataSource(this);
          // buka kontroller
          dataSource.open();

          // ambil semua data barang
          values = dataSource.getAllBarang();

          // masukkan data barang ke array adapter
          ArrayAdapter<Barang> adapter = new ArrayAdapter<Barang>(this,
          android.R.layout.simple_list_item_1, values);

          // set adapter pada list
          setListAdapter(adapter);

          // mengambil listview untuk diset onItemLongClickListener
          ListView lv = (ListView) findViewById(android.R.id.list);
          lv.setOnItemLongClickListener(this);
```

```
    }

    //apabila ada long click
    @Override
    public boolean onItemLongClick(final AdapterView<?> adapter, View v,
int pos,
            final long id) {

        //tampilkan alert dialog
        final Dialog dialog = new Dialog(this);
        dialog.setContentView(R.layout.dialog_view);
        dialog.setTitle("Pilih Aksi");
        dialog.show();
        final Barang b = (Barang) getListAdapter().getItem(pos);
        editButton = (Button) dialog.findViewById(R.id.button_edit_data);
        delButton = (Button) dialog.findViewById(R.id.button_delete_data);

        //apabila tombol edit diklik
        editButton.setOnClickListener(
                new OnClickListener()
                {
                    @Override
                    public void onClick(View v) {
                        // TODO Auto-generated method stub
                        switchToEdit(b.getId());
                        dialog.dismiss();
                    }
                }
                );

        //apabila tombol delete di klik
        delButton.setOnClickListener(
                new OnClickListener()
                {
                    @Override
                    public void onClick(View v) {
                        // Delete barang
                        dataSource.deleteBarang(b.getId());
                        dialog.dismiss();
                        finish();
                        startActivity(getIntent());
                    }
                }
                );

        return true;

    }

    public void switchToEdit(long id)
    {
        Barang b = dataSource.getBarang(id);
        Intent i = new Intent(this, EditData.class);
        Bundle bun = new Bundle();
        bun.putLong("id", b.getId());
        bun.putString("nama", b.getNama_barang());
        bun.putString("merk", b.getMerk_barang());
        bun.putString("harga", b.getHarga_barang());
        i.putExtras(bun);
```

```
        finale();
        startActivity(i);
    }

    public void finale()
    {
        ViewData.this.finish();
        dataSource.close();
    }
    @Override
    protected void onResume() {
        dataSource.open();
        super.onResume();
    }

    @Override
    protected void onPause() {
        dataSource.close();
        super.onPause();
    }
}
```

Sekarang apabila kalian jalankan aplikasinya dan mengklik tombol delete, maka method ***deleteBarang()*** pada controller akan dipanggil, dan data yang dimaksud akan dihapus dari database.

--- Selamat Mencoba ---