Verwendung von Datenbanken unter Android (am Beispiel von gescannten Barcodes)

Hilfsklasse DbHelper:

public class DbHelper extends SQLiteOpenHelper {

private static final String TAG = DbHelper.class.getSimpleName();

// Eckdaten der Datenbank

private static final String DB\_NAME = "SCANS\_DB";

private static final int DB\_VERSION = 1;

private static final String TABLE\_NAME = "scans";

// Bezeichner der Spalten

private static final String KEY\_ID = "\_id";

private static final String KEY\_CONTENT = "\_content";

private static final String KEY\_TIME = "\_time";

private static final String[] COLUMNS = { KEY\_ID, KEY\_CONTENT, KEY\_TIME };

// Konstruktor

public DbHelper(Context context) {

super(context, DB\_NAME, null, DB\_VERSION);

}

Zwei geerbte Methoden müssen überschrieben werden:

@Override

public void onCreate(SQLiteDatabase db) {

String statement = "CREATE TABLE " + TABLE\_NAME + " ( " +

KEY\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +

KEY\_CONTENT +" TEXT, " +

KEY\_TIME + " TEXT );";

try {

db.execSQL(statement);

} catch (android.database.SQLException sqle) {

Log.e(TAG, "error firing sql statement", sqle);

return;

}

}

@Override

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

/\* Hinweis: das tatsächliche Ändern einer Tabelle benötigt tieferes Verständnis von SQL \*/

String statement = "DROP TABLE IF EXISTS " + TABLE\_NAME + ";";

try {

db.execSQL(statement);

} catch (android.database.SQLException sqle) {

Log.e(TAG, "error firing sql statement", sqle);

return;

}

this.onCreate(db);

}

Jetzt können Schnittstellen definiert werden, um mit den Daten zu arbeiten:

public void addScan(BarcodeScanResult scan) {

Log.d(TAG, "addBook -> " + scan.toString());

try {

SQLiteDatabase db = this.getWritableDatabase();

ContentValues values = new ContentValues();

values.put(KEY\_CONTENT, scan.getContent());

values.put(KEY\_TIME, scan.getTime());

db.insert(TABLE\_NAME, null, values);

db.close();

} catch (SQLiteException e) {

Log.e(TAG, "error adding entry to database ", e);

return;

}

}

public BarcodeScanResult getScan(int id) {

SQLiteDatabase db = this.getReadableDatabase();

Cursor cursor =

db.query(TABLE\_NAME,

COLUMNS,

" \_id = ?",

new String[] { String.valueOf(id) },

null, // group by

null, // having

null, // order by

null);// limit

if(cursor!=null) {

cursor.moveToFirst();

}

BarcodeScanResult scan = new BarcodeScanResult();

try {

scan.setId(Integer.parseInt(cursor.getString(0)));

scan.setContent(cursor.getString(1));

scan.setTime(cursor.getString(2));

} catch (NumberFormatException nfe) {

Log.e(TAG, "Error parsing db response", nfe);

return null;

}

Log.d("getScan(" + id + ")", scan.toString());

return scan;

}

public List<BarcodeScanResult> getAllScans() {

List<BarcodeScanResult> scans = new LinkedList<>();

try {

String query = "SELECT \* FROM " + TABLE\_NAME;

SQLiteDatabase db = this.getReadableDatabase();

Cursor cursor = db.rawQuery(query, null);

BarcodeScanResult scan = null;

if(cursor.moveToFirst()) {

do {

scan = new BarcodeScanResult();

scan.setId(Integer.parseInt(cursor.getString(0)));

scan.setContent(cursor.getString(1));

scan.setTime(cursor.getString(2));

scans.add(scan);

} while (cursor.moveToNext());

}

} catch (SQLiteException e) {

Log.e(TAG, "error querying entire database ", e);

} catch (NumberFormatException nfe) {

Log.e(TAG, "Error parsing db response", nfe);

return null;

}

Log.d("getAllScans()",scans.toString());

return scans;

}

public void updateScan(BarcodeScanResult scan) {

// http://hmkcode.com/android-simple-sqlite-database-tutorial/

}

public void deleteScan(BarcodeScanResult scan) {

try {

SQLiteDatabase db = this.getWritableDatabase();

db.delete(TABLE\_NAME,

KEY\_ID + " = ?",

new String[] { String.valueOf(scan.getId())});

db.close();

} catch (SQLiteException e) {

Log.e(TAG, "error querying entire database ", e);

}

Log.d(TAG, "deleteScan() " + scan.toString());

}

Die Klasse BarcodeScanResult:

public class BarcodeScanResult {

private int id;

private String content;

private String time;

public BarcodeScanResult() {

}

public BarcodeScanResult(String content, String time) {

this.content = content;

this.time = time;

}

@Override

public String toString() {

return "Barcode: " + content + " [" + time.toString() + "]";

}

/\* außerdem noch Getter & Setter \*/

Darstellung von Datenbankinhalten mit Hilfe von ListViews und SwipeRefreshLayout

public class HistoryActivity extends ListActivity {

private DbHelper db

private List<BarcodeScanResult> scans = null;

private ArrayAdapter<BarcodeScanResult> adapter;

public ListView listView;

@InjectView(R.id.historyActivity\_swipe\_refresh\_layout)

public SwipeRefreshLayout mSwipeRefreshLayout;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_history);

ButterKnife.inject(this);

db = new DbHelper(this);

**listView = getListView();**

scans = db.getAllScans();

adapter = new ArrayAdapter<BarcodeScanResult>(this,android.R.layout.simple\_list\_item\_1, scans);

setListAdapter(adapter);

mSwipeRefreshLayout.setOnRefreshListener(**new SwipeRefreshLayout.OnRefreshListener()** {

@Override

public void onRefresh() {

refreshContent();

}

});

}

private void refreshContent() {

new Handler().postDelayed(new Runnable() {

@Override

public void run() {

adapter = new ArrayAdapter<BarcodeScanResult>(HistoryActivity.this, android.R.layout.simple\_list\_item\_1, scans);

listView.setAdapter(adapter);

mSwipeRefreshLayout.setRefreshing(false);

}

},0L);

}

Activity\_history.xml

<android.support.v4.widget.SwipeRefreshLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/historyActivity\_swipe\_refresh\_layout">

<ListView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

**android:id="@id/android:list" // Wichtig!**

/>

</android.support.v4.widget.SwipeRefreshLayout>

Navigation Drawer verwenden: (http://hmkcode.com/android-creating-a-navigation-drawer/)

<android.support.v4.widget.DrawerLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@+id/drawer\_layout"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

/\* hier das bisherige Layout der Activity platzieren \*/

<ListView

android:id="@+id/left\_drawer"

android:layout\_width="240dp"

android:layout\_height="match\_parent"

android:layout\_gravity="start"

android:background="#333"

android:choiceMode="singleChoice"

android:divider="#666"

android:dividerHeight="1dp"

android:paddingLeft="15sp"

android:paddingRight="15sp" />

</android.support.v4.widget.DrawerLayout>

In der Activity:

private ListView drawerListView;

private String[] drawerListViewItems;

// am Ende der onCreate Methode

drawerListViewItems = getResources().**getStringArray(R.array.items)**;

drawerListView = (ListView) findViewById(R.id.left\_drawer);

drawerListView.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

switch(position) {

case 0: // erster Menüpunkt

case 1: // zweiter Menüpunkt

default:

}

}

});

drawerListView.setAdapter(new ArrayAdapter<String>(this,

R.layout.listview\_item\_1337, drawerListViewItems));

Listview\_item\_1337.xml (res/layout)

<?xml version="1.0" encoding="utf-8"?>

<TextView xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@android:id/text1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:textColor="#36c922"

android:background="#000"

android:textSize="20sp"

android:gravity="center\_vertical"

android:paddingStart="14.5sp"

android:paddingEnd="14.5sp"

android:minHeight="35sp"

/>