Session 7: Storage

Assignment –Database basics.

Problem Statement

a. What is the use of SQlite open helper class in SQLite?

b. What is the use of OnUpgrade function in SQLiteOpenHelper class?

c. How to show sqlite database table information in android application what is the best way to do it?

Solution:

1. Android provides several ways to store user and app data. SQLite is one way of storing user data. SQLite is a very light weight database which comes with Android OS.

For managing all the operations related to the database , an helper class has been given and is called SQLiteOpenHelper. It automatically manages the creation and update of the database. Its syntax is given below

public class DBHelper extends SQLiteOpenHelper {

public DBHelper(){

super(context,DATABASE\_NAME,null,1);

}

public void onCreate(SQLiteDatabase db) {}

public void onUpgrade(SQLiteDatabase database, int oldVersion, int newVersion) {}

}

One can create own class to handle all database CRUD(Create, Read, Update and Delete) operations using SQLiteOpenHelper. This class makes it easy for ContentProvider implementations to defer opening and upgrading the database until first use, to avoid blocking application startup with long-running database upgrades.

Public constructors:

SQLiteOpenHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int version)

Create a helper object to create, open, and/or manage a database. The database is not actually created or opened until one of getWritableDatabase() or getReadableDatabase() is called.

Parameters:

* context Context: to use to open or create the database
* name String: of the database file, or null for an in-memory database
* factory SQLiteDatabase.CursorFactory: to use for creating cursor objects, or null for the default
* version int: number of the database (starting at 1); if the database is older,
* onUpgrade(SQLiteDatabase, int, int) will be used to upgrade the database; if the database is newer, onDowngrade(SQLiteDatabase, int, int) will be used to downgrade the database
* SQLiteOpenHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int version, DatabaseErrorHandler errorHandler)

Create a helper object to create, open, and/or manage a database. The database is not actually created or opened until one of getWritableDatabase() or getReadableDatabase() is called. Accepts input param: a concrete instance of DatabaseErrorHandler to be used to handle corruption when sqlite reports database corruption.

Parameters:

* context Context: to use to open or create the database
* name String: of the database file, or null for an in-memory database
* factory SQLiteDatabase.CursorFactory: to use for creating cursor objects, or null for the default
* version int: number of the database (starting at 1); if the database is older, onUpgrade(SQLiteDatabase, int, int) will be used to upgrade the database; if the database is newer, onDowngrade(SQLiteDatabase, int, int) will be used to downgrade the database
* errorHandler DatabaseErrorHandler: the DatabaseErrorHandler to be used when sqlite reports database corruption, or null to use the default error handler.

1. OnUpgrade function in SQLiteOpenHelper class is called when the database needs to be upgraded.

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

Called when the database needs to be upgraded. The implementation should use this method to drop tables, add tables, or do anything else it needs to upgrade to the new schema version.

The SQLite ALTER TABLE documentation can be found here. If you add new columns you can use ALTER TABLE to insert them into a live table. If you rename or remove columns you can use ALTER TABLE to rename the old table, then create the new table and then populate the new table with the contents of the old table.This method executes within a transaction. If an exception is thrown, all changes will automatically be rolled back.

Parameters:

db SQLiteDatabase: The database.

oldVersion int: The old database version.

newVersion int: The new database version.

1. Table Layout with cursor.

Showing database information will be well suited with table layout. Since table layout is not an adapter view, you cannot use cursor adapter with it. So, use table layout with cursor to show database table information.

Example:

using the SQLite prepared database from that will get row's and display using the TableLayout with TextView..

database adapter class:

public class DBAdapter3x3 {

public static final String KEY\_ROWID = "\_id";

public static final String KEY\_NAME = "name";

public static final String KEY\_MOVES = "moves";

public static final String KEY\_TIME = "time";

private static final String TAG = "DBAdapter";

private static final String DATABASE\_NAME = "SliderDB3x3.db";

private static final String DATABASE\_TABLE = "topscore3x3";

private static final int DATABASE\_VERSION = 1;

private static final String DATABASE\_CREATE = "create table topscore3x3 " +

"(\_id integer primary key autoincrement, "

+ "name text not null, moves integer not null,"

+ "time text not null);";

private final Context context;

private DatabaseHelper DBHelper;

private SQLiteDatabase db;

public DBAdapter3x3(Context ctx) {

this.context = ctx;

DBHelper = new DatabaseHelper(context);

}

private static class DatabaseHelper extends SQLiteOpenHelper {

DatabaseHelper(Context context) {

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

}

@Override

public void onCreate(SQLiteDatabase db) {

try {

db.execSQL(DATABASE\_CREATE);

} catch (SQLException e) {

e.printStackTrace();

}

}

@Override

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

Log.w(TAG, "Upgrading database from version " + oldVersion + " to "

+ newVersion + ", which will destroy all old data");

db.execSQL("DROP TABLE IF EXISTS contacts");

onCreate(db);

}

}

//---opens the database---

public DBAdapter3x3 open() throws SQLException {

db = DBHelper.getWritableDatabase();

return this;

}

//---closes the database---

public void close() {

DBHelper.close();

}

//---insert a contact into the database---

public long insertContact(String name, int moves,String time) {

ContentValues initialValues = new ContentValues();

initialValues.put(KEY\_NAME, name);

initialValues.put(KEY\_MOVES, moves);

initialValues.put(KEY\_TIME, time);

return db.insert(DATABASE\_TABLE, null, initialValues);

}

//---deletes a particular contact---

public boolean deleteContact(long rowId) {

return db.delete(DATABASE\_TABLE, KEY\_ROWID + "=" + rowId, null) > 0;

}

//---retrieves all the contacts---

public Cursor getAllContacts() {

return db.query(DATABASE\_TABLE, new String[] {KEY\_ROWID, KEY\_NAME,

KEY\_MOVES, KEY\_TIME}, null, null, null, null, null);

}

//---retrieves a particular contact---

public Cursor getContact(long rowId) throws SQLException {

Cursor mCursor = db.query(true, DATABASE\_TABLE, new String[] {KEY\_ROWID, KEY\_NAME,

KEY\_MOVES, KEY\_TIME}, KEY\_ROWID + "=" + rowId, null,null, null, null, null);

if (mCursor != null) {

mCursor.moveToFirst();

}

return mCursor;

}

//---updates a contact---

public boolean updateContact(long rowId, String name, int moves, String time) {

ContentValues args = new ContentValues();

args.put(KEY\_NAME, name);

args.put(KEY\_MOVES, moves);

args.put(KEY\_TIME, time);

return db.update(DATABASE\_TABLE, args, KEY\_ROWID + "=" + rowId, null) > 0;

}

//public Cursor fetchAllNotes() {

public Cursor SortAllRows() {

return db.query(DATABASE\_TABLE, new String[] { KEY\_ROWID, KEY\_NAME,

KEY\_MOVES,KEY\_TIME}, null, null, null, null, KEY\_MOVES + " ASC");

}

}

i used database in this activity

public class TopScore3x3 extends Activity {

private DBAdapter3x3 db;

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.db3x3);

db = new DBAdapter3x3(this);

getall();

delete();

private void delete() {

db.open();

Cursor c = db.SortAllRows();

int i=1;

if (c.moveToFirst()) {

do {

if(i>10) { db.deleteContact(i); }

i++;

} while (c.moveToNext());

}

c.close();

db.close();

}

private void getall() {

//---get all contacts---

db.open();

//db.fetchAllNotes();

Cursor c = db.SortAllRows();

int i=1;

if (c.moveToFirst()) {

do {

DisplayContact(c,i++);

} while (c.moveToNext());

}

c.close();

db.close();

}

public void DisplayContact(Cursor c,int row ) {

String name11 = c.getString(1) + c.getString(2) + c.getString(3);

tv1.setText(name11 );

}

}