

Андройд – Лекц 7

Ж.Золжаргал
(j.zoljargal@csms.edu.mn)



Saving and Loading Files

- `openFileInput`, `openFileOutput` локал файлуудыг унших болон бичих үүсгалаар

```
String FILE_NAME = "tempfile.tmp";
```

```
// Create a new output file stream that's private to this application.  
FileOutputStream fos = openFileOutput(FILE_NAME, Context.MODE_PRIVATE);  
// Create a new file input stream.  
FileInputStream fis = openFileInput(FILE_NAME);
```

- Эдгээр функцууд нь зөвхөн тухайн апп-ын хавтасанд байгаа файлууд руу хандана
- `openFileOutput` функцээр файл үүсгэхэд анхдагч тохиргоо нь өөр апп-с хандах боломжгүй байна.

Saving and Loading Files

- Файлыг олон апп-д хамтран ашиглах стандарт арга нь Content Provider юм
- Үүнтэй адилаар Content Provider ашиглахгүйгээр файлын горимыг дараах байдлаар өөрчлөн ашиглаж болно
 - Context.MODE_WORLD_READABLE
 - Context.MODE_WORLD_WRITEABLE

```
String OUTPUT_FILE = "publicCopy.txt";  
FileOutputStream fos = openFileOutput(OUTPUT_FILE, Context.MODE_WORLD_WRITEABLE);
```

Including Static Files as Resources



- Хэрэв апп-д гадаад файл шаардлагатай тохиолдолд тухайн прожектын res/raw хавтасанд байрлуулж болно.
- Файлын нөөц рүү зөвхөн уншихаар хандах бөгөөд Resources классын openRawResource функцын тусламжтай InputStream-г барьж авна.

```
Resources myResources = getResources();  
InputStream myFile = myResources.openRawResource(R.raw.myfilename);
```

Including Static Files as Resources

- Энэ нь урьдчилан бэлтгэгдсэн өгөгдлийг хадгалахад тохиромжтой. (Толь бичиг)
 - Жишээ нь таны апп олон хэл дэмждэг тохиолдолд
 - Байршилүүд
 - Техникийн тохиргоонууд



File Management Tools

•Андройд нь файл систем дэхь файлуудыг удирдах *File Management Tools*-ээр хангагдсан бөгөөд тэдгээр нь `java.io.File` стандарт пакежид байрладаг.

- `deleteFile` – тухайн апп-аар үүсгэгдсэн файлуудыг устгах

- `fileList` – тухайн апп-р үүсгэгдсэн файлуудын нэрсийг агуулсан `String` төрлийн массив.

- Гэх мэт ...



Databases in Android

- Андройдын бүх баазууд `/data/data/<package_name>/databases` хавтасанд хадгалагддаг.
- Анхдагч тохиргоо нь бүх баазууд хувийн бөгөөд тухайн апп нь хандах болон үүсгэх эрхтэй.
- Апп хооронд өгөгдлийн санг хуваалцахын тулд Content Provider-г хэрэглэх хэрэгтэй.



Introducing SQLite

- SQLite нь хамааралт өгөгдлийн сан.
-(RDBMS – Relational database management system)
- ӨС бүр нь тухайн програмын нэг хэсэг бөгөөд түүгээр үүсгэгддэг. Энэ нь гадаад хамаарлыг багасгаж өгсөн.
- SQLite-д тусад нь Сервер процесс гэж байхгүй
- SQLite нь энгий диск дээрх текст файл руу уншиж бичих үйлдэл хийдэг.
- Олон хүснэгт, индекс, триггер, view-г дэмжих бөгөөд бүгдийг ганц файлд л хадгалдаг.

Cursors and Content Values

- Баазын хүснэгтэд шинэ мөр оруулахад *ContentValues* объектыг хэрэглэдэг
- ContentValues* объект бүр нэг мөрийг баганы нэр болон утга гэсэн бүтэцтэйгээр дүрсэлдэг
- Андройдод *query*-ны үр дүнд *Cursor* объектыг буцаадаг.
- Cursors* нь *query*-ны үр дүн дэх мөрийн дагуу байрлалыг удирддаг.



Cursors and Content Values

- *Cursor* класс нь query-ны үр дүнг удирдах дараах хэдэн аргуудтай
 - *moveToFirst* – Үр дүнгийн эхэнд аваачих
 - *moveToNext* – дараагын мөр рүү шилжих
 - *moveToPrevious* – өмнөх мөр рүү шилжих
 - *getCount* – үр дүн дэхь бичлэгийн тоо
 - *getColumnIndexofThrow* – нэрээр өгсөн баганы индэкс буцаах



Cursors and Content Values

- getColumnName* – *индэксээр өгсөн баганы нэр буцаах*
- getColumnNames* – *бүх баганы нэрийг array хүснэгтээр буцаах*
- moveToPosition* – *заасан мөр рүү курсор шилжүүлэх*
- getPosition* – *одоогын курсорын байрлалыг буцаан*



```
3 import android.database.sqlite.*;
4 import android.database.sqlite.SQLiteDatabase.CursorFactory;
5 import android.util.Log;
6 public class MyDBAdapter {
7     private static final String DATABASE_NAME = "myDatabase.db";
8     private static final String DATABASE_TABLE = "mainTable";
9     private static final int DATABASE_VERSION = 1;
10    // The index (key) column name for use in where clauses.
11    public static final String KEY_ID="_id";
12    // The name and column index of each column in your database.
13    public static final String KEY_NAME="name";
14    public static final int NAME_COLUMN = 1;
15    // TODO: Create public field for each column in your table.
16    // SQL Statement to create a new database.
17    private static final String DATABASE_CREATE = "create table " +
18        DATABASE_TABLE + " (" + KEY_ID +
19        " integer primary key autoincrement, " +
20        KEY_NAME + " text not null);"
21    // Variable to hold the database instance
22    private SQLiteDatabase db;
23    // Context of the application using the database.
24    private final Context context;
25    // Database open/upgrade helper
26    private myDbHelper dbHelper;
27
```

```
28 public MyDBAdapter(Context _context) {
29     context = _context;
30     dbHelper = new myDbHelper(context, DATABASE_NAME, null,
31         DATABASE_VERSION);
32 }
33 public MyDBAdapter open() throws SQLException {
34     db = dbHelper.getWritableDatabase();
35     return this;
36 }
37 public void close() {
38     db.close();
39 }
40 public long insertEntry(MyObject _myObject) {
41     ContentValues contentValues = new ContentValues();
42     // TODO fill in ContentValues to represent the new row
43     return db.insert(DATABASE_TABLE, null, contentValues);
44 }
45 public boolean removeEntry(long _rowIndex) {
46     return db.delete(DATABASE_TABLE, KEY_ID +
47         "=\"" + _rowIndex, null) > 0;
48 }
49 public Cursor getAllEntries () {
50     return db.query(DATABASE_TABLE, new String[] {KEY_ID, KEY_NAME},
51         null, null, null, null, null);
52 }
```

```
53 public MyObject getEntry(long _rowIndex) {
54     MyObject objectInstance = new MyObject();
55     // TODO Return a cursor to a row from the database and
56     // use the values to populate an instance of MyObject
57     return objectInstance;
58 }
59 public int updateEntry(long _rowIndex, MyObject _myObject) {
60     String where = KEY_ID + "=" + _rowIndex;
61     ContentValues contentValues = new ContentValues();
62     // TODO fill in the ContentValues based on the new object
63     return db.update(DATABASE_TABLE, contentValues, where, null);
64 }
65 private static class myDbHelper extends SQLiteOpenHelper {
66     public myDbHelper(Context context, String name,
67         CursorFactory factory, int version) {
68         onCreate(SQLiteDatabase _db) {
69             _db.execSQL(DATABASE_CREATE);
70         }
71         // Called when there is a database version mismatch meaning that
72         // the version of the database on disk needs to be upgraded to
73         // the current version.
74         @Override
75         public void onUpgrade(SQLiteDatabase _db, int _oldVersion,
76             int _newVersion) {
77             // Log the version upgrade.
78             Log.w("TaskDBAdapter", "Upgrading from version " +
```



```
79         _oldVersion + " to " +
80         _newVersion +
81         ", which will destroy all old data");
82     // Upgrade the existing database to conform to the new version.
83     // Multiple previous versions can be handled by comparing
84     // _oldVersion and _newVersion values.
85     // The simplest case is to drop the old table and create a
86     // new one.
87     _db.execSQL("DROP TABLE IF EXISTS " + DATABASE_TABLE);
88     // Create a new one.
89     onCreate(_db);
90 }
91 }
92 }
93 }
```

Using the SQLiteOpenHelper

- *SQLiteOpenHelper* хийсвэр класс бөгөөд баазыг үүсгэх, нээх болон шинэчилэх чухал хэрэглээтэй
- *getReadableDatabase* эсвэл *getWritableDatabase* функцыг дуудсанаар бичих эсвэл унших боломжтой баазын объектыг үүсгэнэ



Using the SQLiteOpenHelper

- *getWritableDatabase* функцыг дуудахад дискний хэмжээ болон хандах эрхийн асуудлаас болж алдаа гарч болзошгүй тул онцгой тохиолдолд *getReadableDatabase* ашиглах хэрэгтэй

```
dbHelper = new myDbHelper(context, DATABASE_NAME, null, DATABASE_VERSION);
```

```
SQLiteDatabase db;
```

```
try {
```

```
    db = dbHelper.getWritableDatabase();
```

```
}
```

```
catch (SQLException ex){
```

```
    db = dbHelper.getReadableDatabase();
```

```
}
```

Using the SQLiteOpenHelper

- Өмнөх код биелэхэд хэрэв бааз үүсээгүй бол *onCreate* event ажиллах бөгөөд харин баазын хувилбар өөрчлөгдсөн бол *onUpgrade* event ажилладаг.



SQLiteHelper классгүй бааз нээх болон үүсгэх

- *SQLiteHelper* классгүй баазыг нээх болон үүсгэх боломжтой бөгөөд үүний тулд *openOrCreateDatabase* функцыг ашиглана.
- *openOrCreateDatabase* функцыг дуудаж шинэ өгөгдлийн сан үүсгэнэ.
- Үүссэн ӨС-ийн объектын *execSQL* функцээр хүснэгт болон тэдгээрийн хамаарлыг үүсгэх *query*-г ажиллуулна

SQLiteHelper классгүй бааз нээх болон үүсгэх

```
private static final String DATABASE_NAME = "myDatabase.db";  
private static final String DATABASE_TABLE = "mainTable";
```

```
private static final String DATABASE_CREATE =  
    "create table " + DATABASE_TABLE +  
    " ( _id integer primary key autoincrement, " +  
    "column_one text not null);";
```

```
SQLiteDatabase myDatabase;
```

```
private void createDatabase() {  
    myDatabase = openOrCreateDatabase(DATABASE_NAME,  
                                       Context.MODE_PRIVATE, null);  
    myDatabase.execSQL(DATABASE_CREATE);  
}
```

Android Database Design Considerations

- Андройд систем дээр өгөгдлийн сангаа зохиомжлоход анхаарах зүйлс
 - ӨС-н хүснэгтэд ихэвч файлуудыг хадгалдаггүй. (зураг, аудио файлууд) Харин замыг нь хадгалах хэрэгтэй
 - Заавал биш ч, Бүх хүснэгт нь auto-increment талбартай буюу мөр бүр unique байх нь чухал. Энэ нь тухайн хүснэгтийг content provider-н тусламжтай хуваалцахад хэрэгтэй.



Querying Your Database

- Database объектын query функцын тусламжтай query-г ажиллуулах бөгөөд дараах параметруудийг дамжуулна.
 - Boolean утга авах бөгөөд үр дүн нь давхардаагүй утгууд байх эсэхийг заана. (optional)
 - Query-г ажиллуулах хүснэгтийн нэр
 - Үр дүнг сонгож авах талбарууд. String төрлийн массив байна.



Querying Your Database

- where нөхцөл үр дүнд ямар мөрүүд сонгогдохыг заана. ? Тэмдэг ашиглаж болох бөгөөд энэ нь сонгогдсон өгөгдлүүдээр солигдох болно
- String төрлийн сонгогдсон өгөгдлүүдийн массив байх бөгөөд энэ нь where нөхцөлийн асуултын тэмдгүүдийн оронд солигдох болно
- Group by – мөрүүдийг хэрхэн групплэхийг заана
- Having – өгөгдлийг шүүх бөгөөд group by-тай хамт хэрэглэгдэнэ

Extracting Results from a Cursor

- Үр дүнгийн Cursor-с өгөгдлөө авахдаа moveTo<location> функцын тусламжтай тухайн мөрийг зөв сонгох хэрэгтэй
- Мөрөө сонгосон бол get<type> функцээр талбарын индексийг дамжуулан талбарын утгыг авах боломжтой

```
String columnValue = myResult.getString(columnIndex);
```

Extracting Results from a Cursor



- Үр дүнгээр давтаж талбарын утгуудын нийлбэрийг олох

```
int GOLD_HOARDED_COLUMN = 2;
Cursor myGold = myDatabase.query("GoldHoards", null, null, null, null,
                                null, null);

float totalHoard = 0f;

// Make sure there is at least one row.
if (myGold.moveToFirst()) {
    // Iterate over each cursor.
    do {
        float hoard = myGold.getFloat(GOLD_HOARDED_COLUMN);
        totalHoard += hoard;
    } while(myGold.moveToNext());
}

float averageHoard = totalHoard / myGold.getCount();
```

Adding, Updating, and Removing Rows

- SQLiteDatabase класс нь insert, delete, update гэсэн функцуудтэй бөгөөд мөн execSQL функцээр гүйцэтгэх боломжтой.
- Баазад шинэчлэлт хийсэн л бол Cursor-н refreshQuery функцыг дуудаж байх хэрэгтэй.

Inserting New Rows

- Шинэ мөр нэмэхэд *ContentValues* объект үүсгэх бөгөөд түүний *put* функцээр багана тус бүрийн утгыг оноож өгнө.
- Харгалцах ӨС-н объектын *insert* функцээр үүсгэсэн *ContentValues* объектыг дамжуулсанаар шинэ мөр баазад үүснэ.



Inserting New Rows



```
// Create a new row of values to insert.
ContentValues newValues = new ContentValues();

// Assign values for each row.
newValues.put(COLUMN_NAME, newValue);
[ ... Repeat for each column ... ]

// Insert the row into your table
myDatabase.insert(DATABASE_TABLE, null, newValues);
```

Updating a Row on the Database

- Insert функцтэй төстэй бөгөөд where нөхцөл авна.

```
// Define the updated row content.
ContentValues updatedValues = new ContentValues();

// Assign values for each row.
updatedValues.put(COLUMN_NAME, newValue);
[ ... Repeat for each column ... ]

String where = KEY_ID + "=" + rowId;

// Update the row with the specified index with the new values.
myDatabase.update(DATABASE_TABLE, updatedValues, where, null);
```

Deleting Rows

- Delete функц энгийн бөгөөд хүснэгтийн нэр болон where нөхцөл авна.

```
myDatabase.delete(DATABASE_TABLE, KEY_ID + "=" + rowId, null);
```

