G54MDP Lab Session 04 - Storage

The aim of this exercise is for you to implement two simple applications that make use of Android storage component in two different ways.

Necessarily this will be similar to the example application shown in lectures 9/10, so spend some time reviewing the code provided, however it is important to try and implement a similar application yourself rather than cutting and pasting code, so that you get used to the coding style required.

Shared Preferences

Create an application that makes use of SharedPreferences to store a single String variable.

Create a new Activity in the usual way, with a simple View containing a text field, a Button and an onClick handler method. Make use of the SharedPreferences API to retrieve the contents of the text field and write it to the Application's shared preferences.

Inspect the private data storage of the Application on the device using the **adb shell** command, looking in /data/data/your_app_classname/

Extend the Activity's onCreate method to retrieve the stored data and write it to the text field when the application is started.

Reading preferences:

```
SharedPreferences settings  = getSharedPreferences(“myPreferences”, 0);

String pref = settings.getString(“myPreference1”, "defaultValue");
```

Writing preferences:

```
SharedPreferences settings = getSharedPreferences(“myPreferences”, 0);
SharedPreferences.Editor editor = settings.edit();
editor.putString(“myPreference1”, myString);
editor.commit();
```

Database Storage

Create an application that stores structured data in a SQLite database

Extend your application to make use of a database to store multiple entries from a text field, and then display all entries back to the user via a second text field.

Begin by creating a database helper class that extends SQLiteOpenHelper. As per the example given, extend the onCreate method to create a simple table within a database, using the execSQL method.

```
public void onCreate(SQLiteDatabase db)
{
	db.execSQL("CREATE TABLE myList (" +
				"id INTEGER PRIMARY KEY AUTOINCREMENT ," +
				"name VARCHAR(128) NOT NULL" +
				");");
```

Next, make use of the database from within the Activity, to insert new rows into the database, and query the database to retrieve existing rows to display. You'll need to instantiate your database helper class, and use it to retrieve a handle to the writeable database.

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

…

db.execSQL("INSERT INTO myList (name) " +
                            "VALUES " +
                            "('" + input + "');");
```

To retrieve rows you will need to make use of a cursor object to move across the results:

```
Cursor c = db.query("myList", new String[] { "id", "name" }, null, null,
null, null, null);

if(c.moveToFirst())
{
	int id = c.getInt(0);
	String name = c.getString(1);

…
```

Look at the documentation to use methods that are appropriate to the particular table that you have created.

http://developer.android.com/guide/topics/data/data-storage.html

http://developer.android.com/reference/android/database/sqlite/SQLiteDatabase.html

http://developer.android.com/reference/android/database/Cursor.html