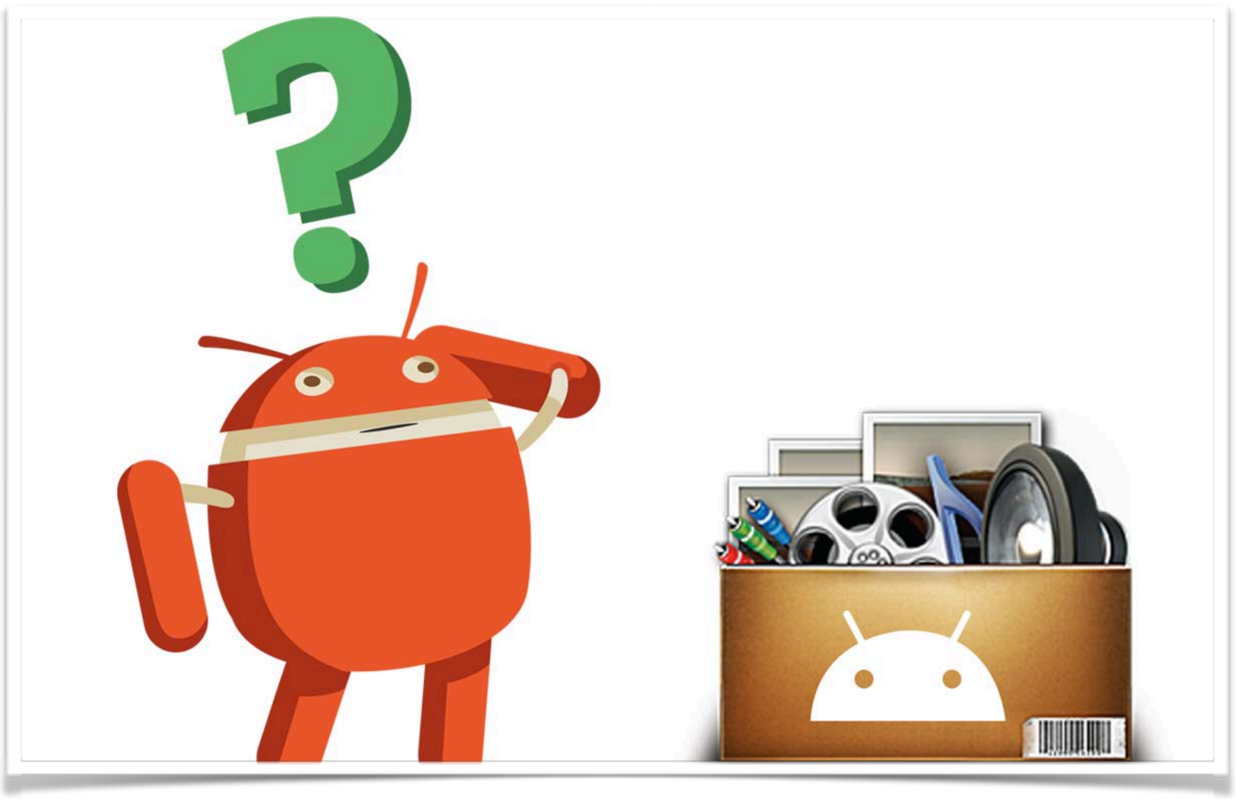
CS425 - Web and Mobile Software Engineering
Dr. Lilia Sfaxi
27 avril 2017

MedTech

Storage

Android Lab 04



I. Storage in Android

There are several options to store persistent data in Android. The choice of the ideal solution depends on the specific needs of the users:

- Is the data private or accessible to other applications?
- How much space is required?
- Must the data be structures, semi-structured or not structured?

The storage options we present in this lab are the following:

1. Shared Preferences

It is a general framework that enables to save and extract key-value pairs of primitive type. It enables to store and retrieve boolean, float, integer, long and string data.

Typically, the Shared Preferences are used to save the preferences of the users, such as the ring.

2. Internal Storage

It is possible to use the internal storage of the device to save files. By default, those files are private (inaccessible to other applications). When the application is deleted, these files are too.

3. External Storage

All Android compatible devices support external storage, that can be an SD card, or an internal non-removable space.

Saved files in an external storage are accessible to all applications, in read-only mode. They can be modified by the user if the « USB mass storage » is activated to transfer the files to a computer.

4. SQLite Database

Android provides a total support for SQLite DBMS. SQLite is a software library implementing a SQL database engine with zero-configuration, very light and without external dependencies.

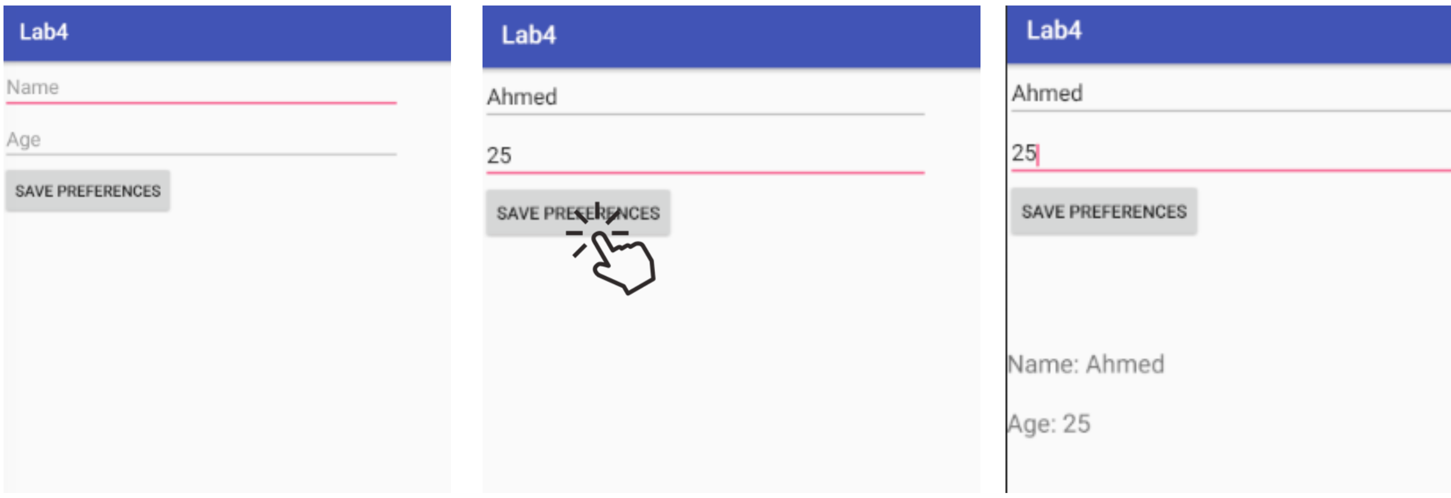
All databases created in an application are accessible by name from everywhere in this application, but not from the outside of the application.

A recommended method to create a SQLite database is to use a sub-class of SQLiteOpenHelper, a helper class that facilitates the creation and handling of SQLite databases.

II. Exercise 1: Shared Preferences

1. Objective

The goal of this first part is to create a simple application to store some data in the Shared Preferences. The graphical interface will be as follows:



2. Implementation

To read an object from Shared Preferences, use one of the following methods:

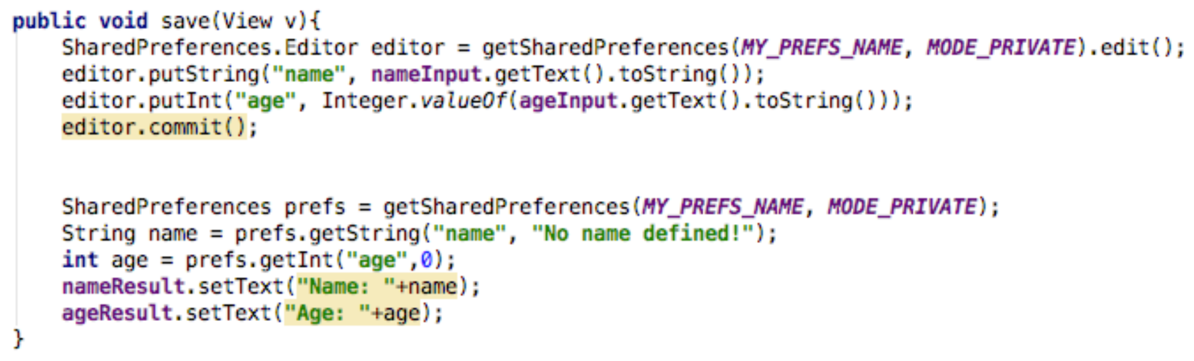
- *getSharedPreferences*: to use several preference files defined by name.
- *getPreferences*: to use a single preference file, without defining a file name.

To add values:

- Call *edit()* to get a SharedPreferences.Editor
- Add values with methods such as: *putBoolean()* and *putString()*
- Validate these values with *commit()*

To read values, use *getBoolean()*, *getString()*,…

The obtained code looks like the following:



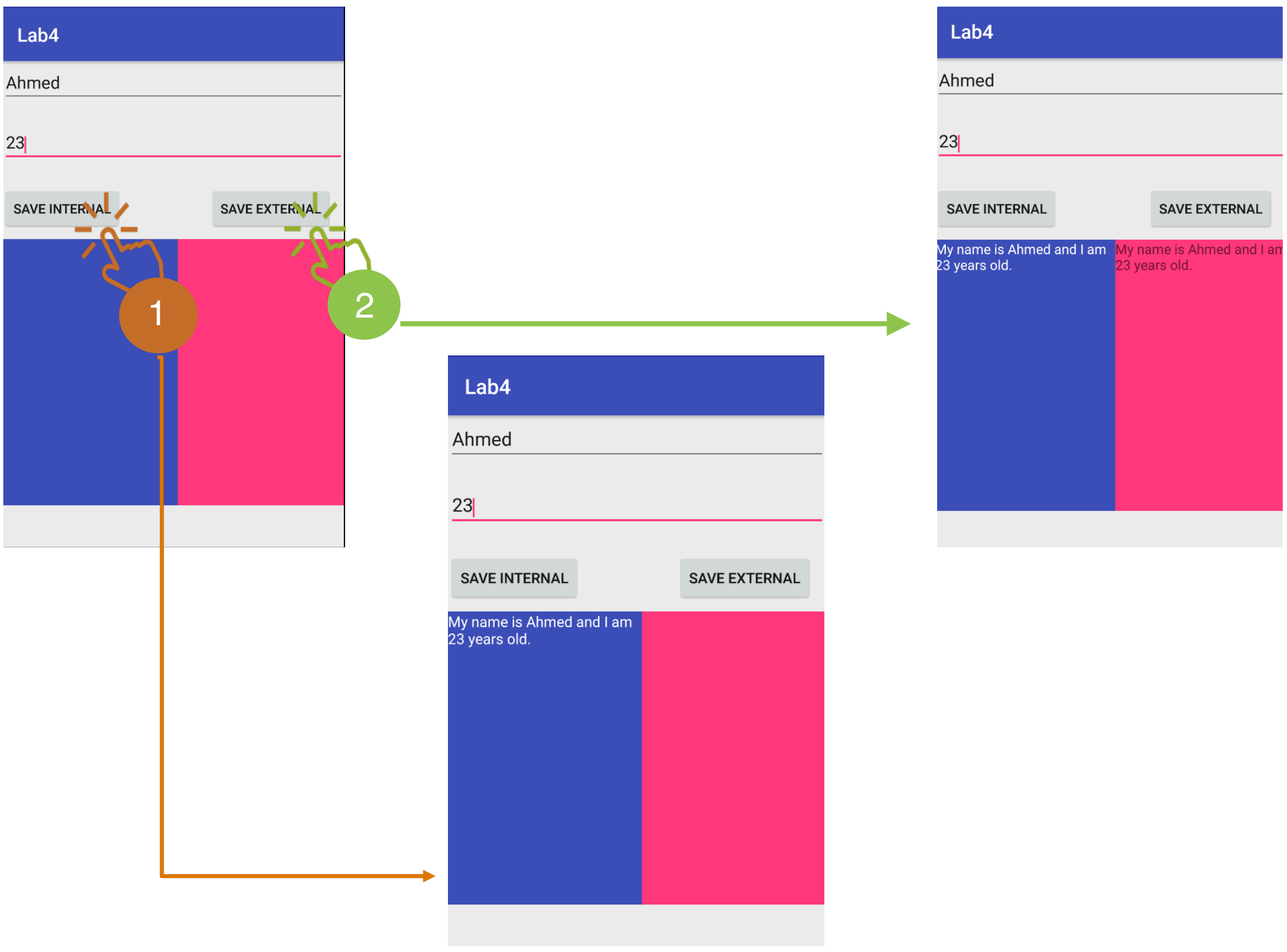
```java
public void save(View v){
    SharedPreferences.Editor editor = getSharedPreferences(MY_PREFS_NAME, MODE_PRIVATE).edit();
    editor.putString("name", nameInput.getText().toString());
    editor.putInt("age", Integer.valueOf(ageInput.getText().toString()));
    editor.commit();


    SharedPreferences prefs = getSharedPreferences(MY_PREFS_NAME, MODE_PRIVATE);
    String name = prefs.getString("name", "No name defined!");
    int age = prefs.getInt("age",0);
    nameResult.setText("Name: "+name);
    ageResult.setText("Age: "+age);
}
```

Activity 1: Create a project called Lab4, enabling to save and read data in the Shared Preferences.

III. Exercise 2: Internal and External Storage

1. Objective

The main objective of this part is to create a simple application to store some data in internal and external files. The interface will look like the following:

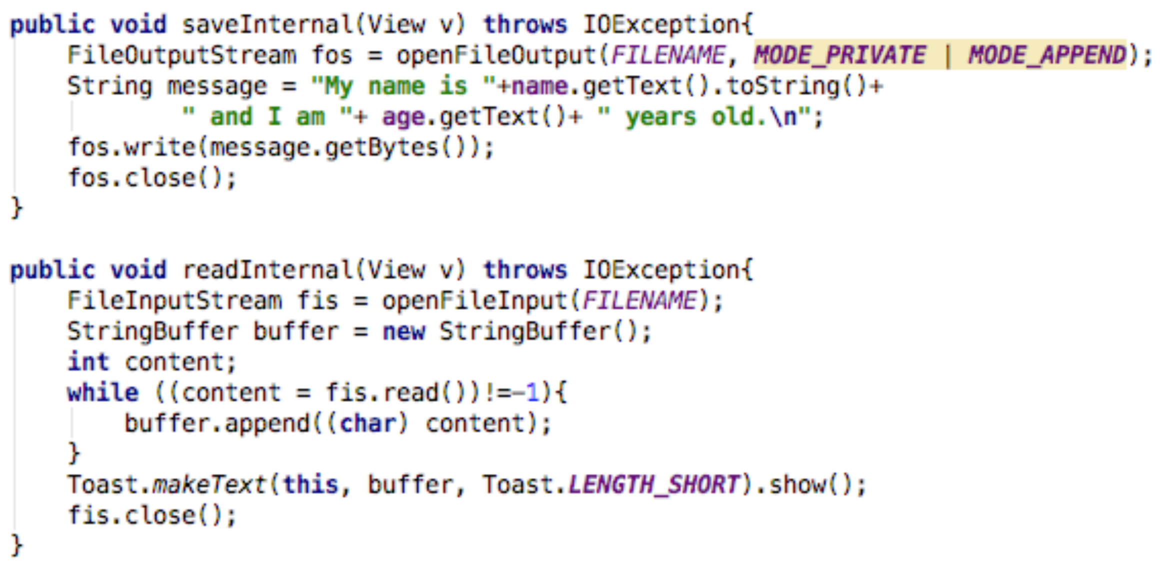

Lab4
Ahmed
23
SAVE INTERNAL
SAVE EXTERNAL
1
2
Lab4
Ahmed
23
SAVE INTERNAL
SAVE EXTERNAL
My name is Ahmed and I am 23 years old.
My name is Ahmed and I am 23 years old.
Lab4
Ahmed
23
SAVE INTERNAL
SAVE EXTERNAL
My name is Ahmed and I am 23 years old.

2. Internal Storage

To read and write in a file internal to the application, use the Java primitives *FileInputStream* and *FileOutputStream.* When writing data, it is mandatory to specify one (or many) operational mode(s). These modes can be combined using | operator.

- MODE_PRIVATE : The file is accessible only for the application that created it.
- MODE_WORLD_READABLE : The file is accessible by other applications in read-only.
- MODE_WORLD_WRITEABLE : The file is writable by other applications.
- MODE_APPEND : If the file already exists, the data is appended at the end.

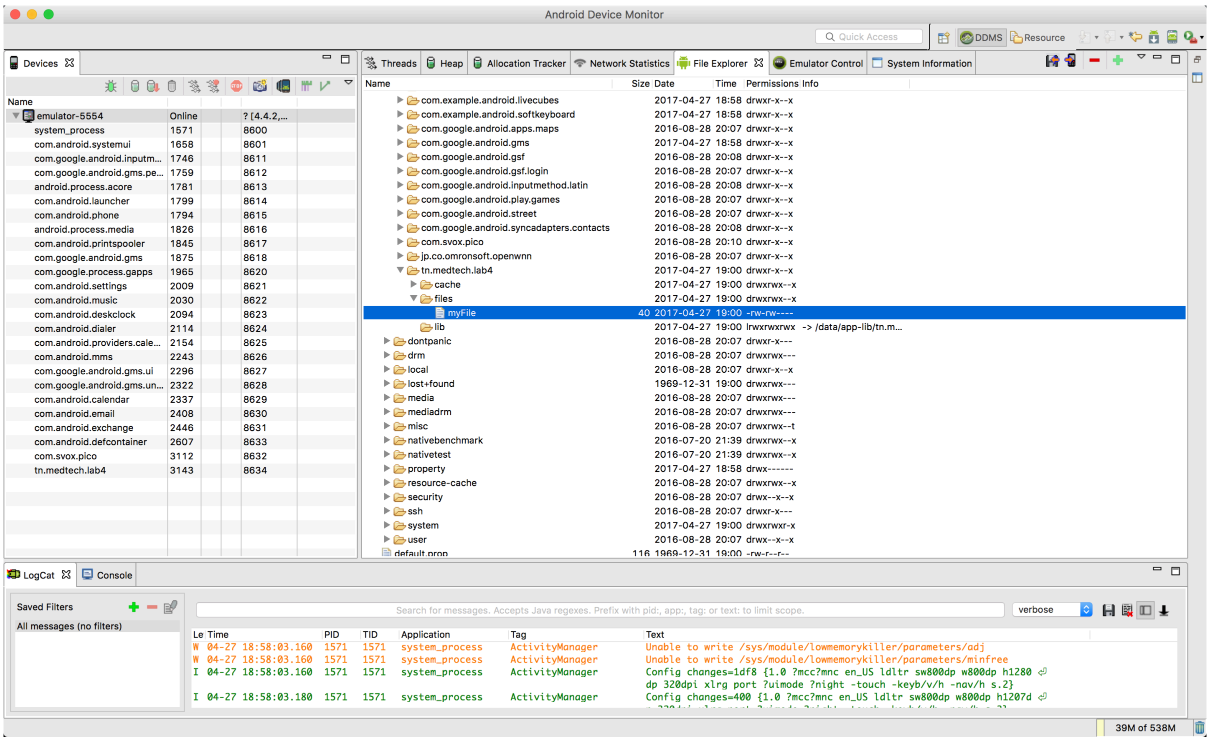
Here are some examples to read and write internal files.



```java
public void saveInternal(View v) throws IOException{
    FileOutputStream fos = openFileOutput(FILENAME, MODE_PRIVATE | MODE_APPEND);
    String message = "My name is "+name.getText().toString()+
            " and I am "+ age.getText()+ " years old.\n";
    fos.write(message.getBytes());
    fos.close();
}

public void readInternal(View v) throws IOException{
    FileInputStream fis = openFileInput(FILENAME);
    StringBuffer buffer = new StringBuffer();
    int content;
    while ((content = fis.read())!=-1){
        buffer.append((char) content);
    }
    Toast.makeText(this, buffer, Toast.LENGTH_SHORT).show();
    fis.close();
}
```

To display the files of your application, open the Android Device Monitor (under Tools -> Android) and choose *File Explorer*[1]. You can find the file you created under le directory /data/data/<your project package>/files

[1] There is a bug in the API 24 devices preventing the file explorer to be displayed. Use devices with lower API for testing.



Android Device Monitor

```
Devices
Name
emulator-5554                      Online      ? [4.4.2,...
  system_process                   1571        8600
  com.android.systemui             1658        8601
  com.google.android.inputm...     1746        8611
  com.google.android.gms.pe...     1759        8612
  android.process.acore            1781        8613
  com.android.launcher             1799        8614
  com.android.phone                1794        8615
  android.process.media            1826        8616
  com.android.printspooler         1845        8617
  com.google.android.gms           1875        8618
  com.google.process.gapps         1965        8620
  com.android.settings             2009        8621
  com.android.music                2030        8622
  com.android.deskclock            2094        8623
  com.android.dialer               2114        8624
  com.android.providers.cale...    2154        8625
  com.android.mms                  2243        8626
  com.google.android.gms.ui        2296        8627
  com.google.android.gms.un...     2322        8628
  com.android.calendar             2337        8629
  com.android.email                2408        8630
  com.android.exchange             2446        8631
  com.android.defcontainer         2607        8633
  com.svox.pico                    3112        8632
  tn.medtech.lab4                  3143        8634
```

```
File Explorer
Name                                          Size Date       Time  Permissions Info
    com.example.android.livecubes                  2017-04-27 18:58 drwxr-x--x
    com.example.android.softkeyboard               2017-04-27 18:58 drwxr-x--x
    com.google.android.apps.maps                   2016-08-28 20:07 drwxr-x--x
    com.google.android.gms                         2017-04-27 18:58 drwxr-x--x
    com.google.android.gsf                         2016-08-28 20:08 drwxr-x--x
    com.google.android.gsf.login                   2016-08-28 20:07 drwxr-x--x
    com.google.android.inputmethod.latin           2016-08-28 20:08 drwxr-x--x
    com.google.android.play.games                  2016-08-28 20:07 drwxr-x--x
    com.google.android.street                      2016-08-28 20:07 drwxr-x--x
    com.google.android.syncadapters.contacts       2016-08-28 20:08 drwxr-x--x
    com.svox.pico                                  2016-08-28 20:10 drwxr-x--x
    jp.co.omronsoft.openwnn                        2016-08-28 20:07 drwxr-x--x
    tn.medtech.lab4                                2017-04-27 19:00 drwxr-x--x
      cache                                        2017-04-27 19:00 drwxrwx--x
      files                                        2017-04-27 19:00 drwxrwx--x
        myFile                                  40 2017-04-27 19:00 -rw-rw----
      lib                                          2017-04-27 19:00 lrwxrwxrwx  -> /data/app-lib/tn.m...
  dontpanic                                        2016-08-28 20:07 drwxr-x---
  drm                                              2016-08-28 20:07 drwxrwx---
  local                                            2016-08-28 20:07 drwxr-x--x
  lost+found                                       1969-12-31 19:00 drwxrwx---
  media                                            2016-08-28 20:07 drwxrwx---
  mediadrm                                         2016-08-28 20:07 drwxrwx---
  misc                                             2016-08-28 20:07 drwxrwx--t
  nativebenchmark                                  2016-07-20 21:39 drwxrwx--x
  nativetest                                       2016-07-20 21:39 drwxrwx--x
  property                                         2017-04-27 18:58 drwx------
  resource-cache                                   2016-08-28 20:07 drwxrwx--x
  security                                         2016-08-28 20:07 drwx--x--x
  ssh                                              2016-08-28 20:07 drwxr-x---
  system                                           2017-04-27 19:00 drwxrwxr-x
  user                                             2016-08-28 20:07 drwx--x--x
default.prop                                   116 1969-12-31 19:00 -rw-r--r--
```

```
LogCat
Saved Filters
All messages (no filters)

Le Time                PID   TID   Application     Tag              Text
W  04-27 18:58:03.160  1571  1571  system_process  ActivityManager  Unable to write /sys/module/lowmemorykiller/parameters/adj
W  04-27 18:58:03.160  1571  1571  system_process  ActivityManager  Unable to write /sys/module/lowmemorykiller/parameters/minfree
I  04-27 18:58:03.160  1571  1571  system_process  ActivityManager  Config changes=1df8 {1.0 ?mcc?mnc en_US ldltr sw800dp w800dp h1280
                                                                    dp 320dpi xlrg port ?uimode ?night -touch -keyb/v/h -nav/h s.2}
I  04-27 18:58:03.180  1571  1571  system_process  ActivityManager  Config changes=400 {1.0 ?mcc?mnc en_US ldltr sw800dp w800dp h1207d
```

Activity 2:

-Create the interface in a new activity, use an intent to go from the preceding activity (Shared Preferences) to this one.

-Test the behaviour of the methods *readInternal* and *saveInternal* and display the result in a toast, for now.

3. External Storage

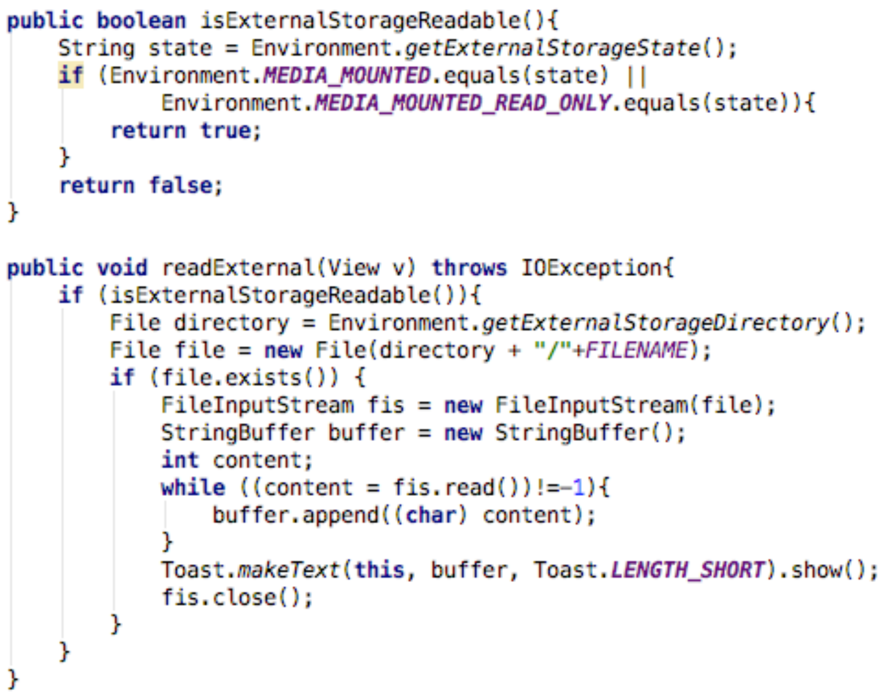
To read or write files on the external storage, the application must have READ_EXTERNAL_STORAGE and WRITE_EXTERNAL_STORAGE permissions.

Before manipulating the storage, il must first verify its availability, using the method *getExternalStorageState*.

Example of writing:

```java
public boolean isExternalStorageWritable(){
    String state = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(state)){
        return true;
    }
    return false;
}

public void saveExternal(View v) throws IOException{
    if (isExternalStorageWritable()){
        File directory = Environment.getExternalStorageDirectory();
        File file = new File(directory + "/"+FILENAME);
        FileOutputStream fos = new FileOutputStream(file);
        String message = "My name is "+name.getText().toString()+
                " and I am "+ age.getText()+ " years old.\n";
        fos.write(message.getBytes());
        fos.close();
    }
}
```

Example of reading:



```java
public boolean isExternalStorageReadable(){
    String state = Environment.getExternalStorageState();
    if (Environment.MEDIA_MOUNTED.equals(state) ||
            Environment.MEDIA_MOUNTED_READ_ONLY.equals(state)){
        return true;
    }
    return false;
}

public void readExternal(View v) throws IOException{
    if (isExternalStorageReadable()){
        File directory = Environment.getExternalStorageDirectory();
        File file = new File(directory + "/"+FILENAME);
        if (file.exists()) {
            FileInputStream fis = new FileInputStream(file);
            StringBuffer buffer = new StringBuffer();
            int content;
            while ((content = fis.read())!=-1){
                buffer.append((char) content);
            }
            Toast.makeText(this, buffer, Toast.LENGTH_SHORT).show();
            fis.close();
        }
    }
}
```

Activity 3: Add the part where you write in an external file. Change the layout in order to display the read text in the corresponding TextArea (as showed in the interface above).

III. SQLite Database

1. Objective

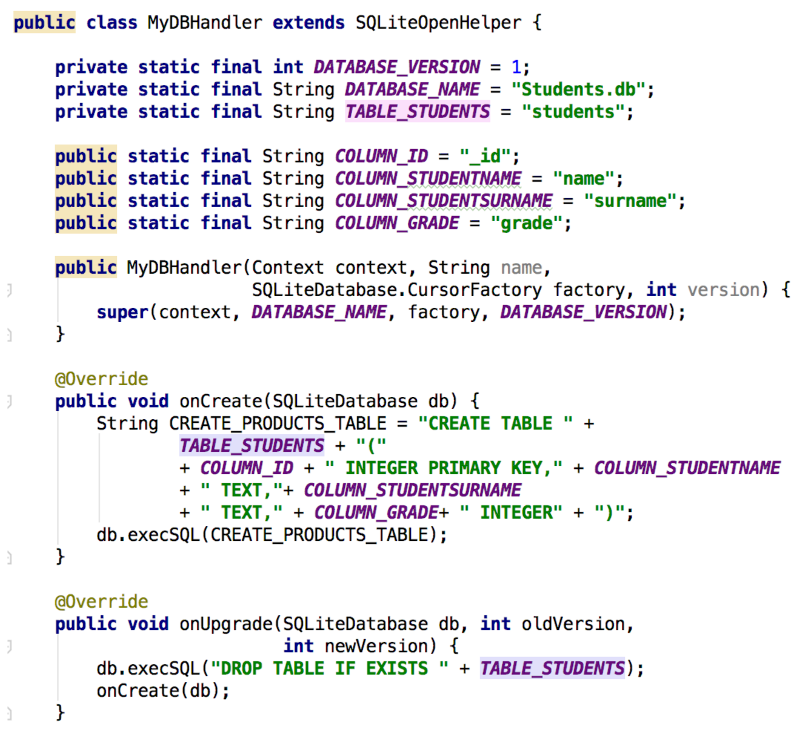
Create simple application enabling to handle the elements of a database. The operations you have to implement are: insertion, display and deletion of element.

2. Database creation

To handle a SQLite database, the recommended steps are the following:

1. Create a class as a model for each table of the database
2. Create a class inheriting from SQLiteOpenHelper
3. Define a schema for your database to declare how the data is to be organized. To create a schema, a good way is to create a static class implementing the BaseColumns interface. When doing this, your inner class is guaranteed to work harmoniously with the Android Framework. It is though not a necessity.
4. Implement the following methods:
    1. The constructor
    2. onCreate: contains the operations at the creation of the database
    3. onUpgrade: contains the operations at the upgrade of the database.

The obtained SQLiteOpenHelper class looks like the following:



```java
public class MyDBHandler extends SQLiteOpenHelper {

    private static final int DATABASE_VERSION = 1;
    private static final String DATABASE_NAME = "Students.db";
    private static final String TABLE_STUDENTS = "students";

    public static final String COLUMN_ID = "_id";
    public static final String COLUMN_STUDENTNAME = "name";
    public static final String COLUMN_STUDENTSURNAME = "surname";
    public static final String COLUMN_GRADE = "grade";

    public MyDBHandler(Context context, String name,
                       SQLiteDatabase.CursorFactory factory, int version) {
        super(context, DATABASE_NAME, factory, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        String CREATE_PRODUCTS_TABLE = "CREATE TABLE " +
                TABLE_STUDENTS + "("
                + COLUMN_ID + " INTEGER PRIMARY KEY," + COLUMN_STUDENTNAME
                + " TEXT,"+ COLUMN_STUDENTSURNAME
                + " TEXT," + COLUMN_GRADE+ " INTEGER" + ")";
        db.execSQL(CREATE_PRODUCTS_TABLE);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion,
                          int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_STUDENTS);
        onCreate(db);
    }
```

3. Record Insertion

To insert an element in the database, use the object ContentValues, that contains, for each record, the values of the different columns.

```
ContentValues values = new ContentValues() ;
values.put(<column1_name>,<column1_value>) ;
values.put(<column2_name>,<column2_value>) ;
```

Once the record created, insert it in the database using an SQLiteDatabase object. This object is obtained by calling *this.getWritableDatabase()* from the SQLiteOpenHelper class.

```
SQliteDatabase db = this.getWritableDatabase();
db.insert(<table_name>,null,values) ;
```

Il ne faut jamais oublier de fermer la table une fois l'opération terminée.

```
db.close();
```

4. Record Lookup

To lookup and display an element of a table

1. Create a string containing the SQL SELECT query you want to execute.
2. Create a cursor to go through the result of the query:

```
Cursor cursor = db.rawQuery(query, null) ;
if (cursor.moveToFirst()){
    cursor.moveToFirst() ;
    int elem1 = Integer.parseInt(cursor.getString(0)) ;
    String elem2 = cursor.getString(1) ;
    …
    cursor.close() ;
}
db.close() ;
```

5. Deletion

To delete one or several elements from the database:

1. Create a string containing the SQL SELECT query you want to execute.
2. Create a cursor to go through the result.
3. Use the following function:

```
db.delete(<table_name>, <id_name> + " = ? ",
      <id_value>) ;
```

4. Don't forget to close the cursor and the database!