

TD6 – SQLite pour Android

Sommaire :

Introduction.....	2
• Objectif du TP	
Réalisation.....	3-12
1. Classe Contact	
2. Database Helper	
3. Gestion de la BDD	
4. Manipulation de la BDD	
5. Activité menu	
6. Initialisation de la BDD avec des Boutons	
7. Affichage de la liste des contacts	
8. Ajout d'un contact	
9. Modification d'un contact ajouté précédemment	
10. Supression d'un contact demandé	
Conclusion.....	13

Introduction :

On veut, dans ce TP, construire une application Android qui permet de gérer des contacts (essentiellement un nom associé à un numéro de téléphone).

Ces contacts seront mis dans une base de données Android gérée par SQLite.

Construction de la base de données

```
public Contact(int _id, String nom, String numTelephone) {  
    this._id = _id;  
    this.nom = nom;  
    this.numTelephone = numTelephone;  
}
```

Compléter cette classe avec des accesseurs et des constructeurs appropriés.

```
public int get_id() { return _id; }  
  
public void set_id(int _id) {  
    this._id = _id;  
}  
  
public String getNom() { return nom; }  
  
public void setNom(String nom) { this.nom = nom; }  
  
public String getNumTelephone() { return numTelephone; }  
  
public void setNumTelephone(String numTelephone) { this.numTelephone = numTelephone; }
```

2°) Construire un "Database Helper" permettant de gérer une base de données. Le début de cette classe est :

```
public class DataBaseHelper extends SQLiteOpenHelper {

    // All Static variables
    // Database Version
    private static final int DATABASE_VERSION = 1;
    // Database Name
    private static final String DATABASE_NAME = "contactsManager";
    // Contacts table name
    private static final String TABLE_CONTACTS = "contacts";
    // Contacts Table Columns names
    private static final String KEY_ID = "id";
    private static final String KEY_NAME = "name";
    private static final String KEY_PH_NO = "phone_number";

    private static final String REQUETE_CREATION_TABLE = "create table "
        + TABLE_CONTACTS + " (" + KEY_ID
        + " INTEGER PRIMARY KEY AUTOINCREMENT, " + KEY_NAME
        + " text not null, " + KEY_PH_NO + " text not null);";

    public DataBaseHelper(Context context) { super(context,DATABASE_NAME, factory: null,DATABASE_VERSION); }
```

3°) Compléter cette classe de sorte à :

- a) Créer la base de données
- b) Pouvoir insérer des Contacts dans cette base
- c) Récupérer tous les contacts de la base

```
@Override
public void onCreate(SQLiteDatabase db) { db.execSQL(REQUETE_CREATION_TABLE); }

@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("drop TABLE " + TABLE_CONTACTS + ";");
    onCreate(db);
}
```

```
public long insertContact(Contact contact) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues valeurs = new ContentValues();
    valeurs.put(KEY_NAME, contact.getNom());
    valeurs.put(KEY_PH_NO, contact.getNumTelephone());
    return db.insert(TABLE_CONTACTS, nullColumnHack: null, valeurs);
}

@Deprecated
public Contact getContact(int id){
    SQLiteDatabase db = this.getWritableDatabase();
    db.query(TABLE_CONTACTS, columns: null, selection: null, selectionArgs: null, groupBy: null, having: null, orderBy: null);
    return null;
}

public List<Contact> getAllContacts() {
    List<Contact> list = new ArrayList<>();
    SQLiteDatabase db = this.getWritableDatabase();
    Cursor cursor = db.query(TABLE_CONTACTS, columns: null, selection: null, selectionArgs: null, groupBy: null, having: null, orderBy: null);
    if(!cursor.moveToFirst()){
        return list;
    }
    do {
        list.add(new Contact(cursor.getInt( columnIndex: 0),cursor.getString( columnIndex: 1),cursor.getString( columnIndex: 2)));
    } while (cursor.moveToNext());
    return list;
}
```

4°) Manipuler cette base à l'aide d'une activité principale possédant la méthode :

```
public void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    LeDatabaseHandler db = new LeDatabaseHandler(this);

    /*
     * Operation CRUD
     */
    Log.d("JMF", "Insertion de Contact");
    db.addContact(new Contact("Jo", "9100000000"));
    db.addContact(new Contact("Jack", "9199999999"));
    db.addContact(new Contact("William", "9522222222"));
    db.addContact(new Contact("Averel", "9533333333"));

    // Reading all contacts
    Log.d("JMF", "Lecture des Contacts");
    List<Contact> contacts = db.getAllContacts();

    for (Contact cn : contacts) {
        String log = "Id: "+cn.getID()+" ,Name: " + cn.getName() + " ,Phone: "
+ cn.getPhoneNumber();
        // Writing Contacts to
        log Log.d("JMF", log);
    }
}
```

5°) Définir une activity qui fait afficher :



```
<TextView
    android:id="@+id/textView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Gestion de contact - SQLITE"
    android:textSize="30sp" />

<Button
    android:id="@+id/btn_menu_1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Afficher tous vos contacts" />

<Button
    android:id="@+id/btn_menu_2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Ajouter un contact" />

<Button
    android:id="@+id/btn_menu_3"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Modifier un contact" />

<Button
    android:id="@+id/btn_menu_4"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Vider et réinitialiser la BDD" />
```

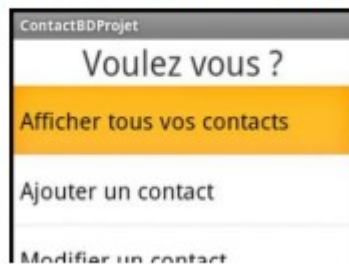
6°) Ajouter le code de sorte que lorsque l'utilisateur clique sur l'item "Initialisation de la base !", la base (table) est recrée avec 4 Contacts.

On rappelle que le code de gestion d'une ListView est :

```
lv.setOnItemClickListener(new OnItemClickListener() {  
    public void onItemClick(AdapterView<?> parent, View view,  
        int position, long id) {  
        if (position == 0) {  
            // traitement si l'utilisateur a choisi le premier item de la ListView  
        } else { ...  
        }  
    }  
});
```

```
public class MainActivity extends AppCompatActivity implements AdapterView.OnItemClickListener, View.OnClickListener {  
  
    private transient ContactDB_DAO contactDB_dao;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
  
        super.onCreate(savedInstanceState);  
  
        setContentView(R.layout.activity_main);  
  
        DataBaseHelper leHelper = new DataBaseHelper( context: this);  
        contactDB_dao = new ContactDB_DAO( ctx: this);  
        //SQLiteDatabase maBaseDonnees = leHelper.getWritableDatabase();  
  
        Contact contact = new Contact( _id: 42, nom: "ANDRE", numTelephone: "0768683537");  
        contactDB_dao.insertContact(contact);  
        Button button = (Button) findViewById(R.id.btn_menu_1);  
        Button button4 = (Button) findViewById(R.id.btn_menu_4);  
        Button button2 = (Button) findViewById(R.id.btn_menu_2);  
        Button button3 = (Button) findViewById(R.id.btn_menu_3);  
  
        button.setOnClickListener(this);  
        button2.setOnClickListener(this);  
        button3.setOnClickListener(this);  
        button4.setOnClickListener(this);  
  
    }  
}
```

7°) De même, écrivez le code qui affiche tous les ontacts lorsque l'utilisateur a choisi le premier item. On pourra utiliser un `TableLayout`.



```
private void updateList(){
    ContactDB_DAO contactDB_dao = new ContactDB_DAO( ctx: this);
    list.clear();
    list.addAll(contactDB_dao.getAllContacts());
    contactDB_dao.close();
    arrayAdapter.notifyDataSetChanged();
}

@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (resultCode == 2) {
        updateList();
        // arrayAdapter.notifyDataSetChanged();
        return;
    }
}

@Override
public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
    Contact contact = list.get(position);
    if(edit){
        Intent intent = new Intent( packageContext: this,EditContact.class);
        intent.putExtra( name: "name",contact.getNom());
        intent.putExtra( name: "num",contact.getNumTelephone());
        intent.putExtra( name: "id",contact.get_id());
        intent.putExtra( name: "edit",edit);
        startActivityForResult(intent,RESULT_CANCELED);
    }
}
```

8°) Ecrire une activity qui affiche :



ContactBDProjet

Nom :

Numéro de téléphone :

Ajouter ce contact

```
<LinearLayout
    android:id="@+id/linearLayout2"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:orientation="vertical"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <TextView
            android:id="@+id/txtv_edit_name"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Nom : " />

        <EditText
            android:id="@+id/edittxt_edit_name"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:ems="10"
            android:inputType="textPersonName"
            android:text="" />

    </LinearLayout>
```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">


    <TextView
        android:id="@+id/txtv_edit_num"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="TextView" />

    <EditText
        android:id="@+id/edittxt_edit_num"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:ems="10"
        android:inputType="textPersonName"
        android:text="" />

</LinearLayout>

<Button
    android:id="@+id/btn_edit"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Valider" />

</LinearLayout>
```

Cette activité est lancée lorsque l'utilisateur sélectionne l'item "Ajouter un contact".
Écrire le code permettant d'ajouter un Contact dans la base de données grâce à cette interface.

9°) Ecrire l'activité qui présente tous les contacts dans une `ListView` pour pouvoir en modifier. L'enchaînement des activités doit être :

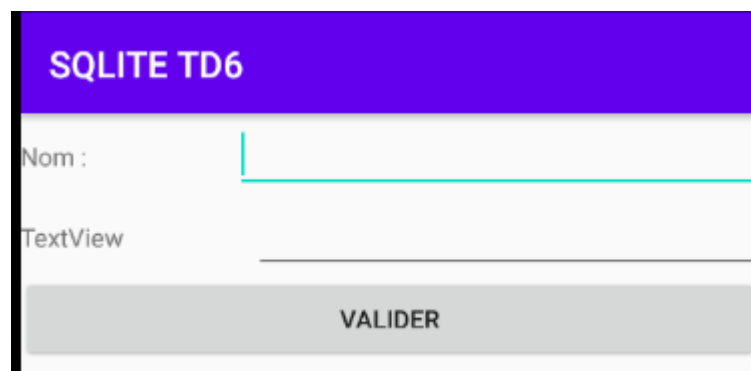


```
@Override
public void onClick(View v) {
    EditText editTextName = findViewById(R.id.edittxt_edit_name);
    EditText editTextNum = findViewById(R.id.editxt_edit_num);
    String name = editTextName.getText().toString();
    String num = editTextNum.getText().toString();
    ContactDB_DAO contactDB_dao = new ContactDB_DAO( ctx: this);
    int id = getIntent().getIntExtra( name: "id", defaultValue: -1);
    Contact contact = new Contact(id,name,num);
    if(edit){
        contactDB_dao.updateContact(contact);
        contactDB_dao.close();
        setResult(2);
    } else {
        contactDB_dao.insertContact(contact);
        setResult(3);
    }
    contactDB_dao.close();
    finish();
}
```

```
@Override
public void onClick(View v) {

    Log.i( tag: "TAGTAGTAG", msg: "btn");
    switch (v.getId()){
        case R.id.btn_menu_1:
            Intent i = new Intent( packageContext: this, ListContact.class);
            i.putExtra( name: "edit", value: false);
            startActivity(i);
            break;
        case R.id.btn_menu_2:
            Intent i2 = new Intent( packageContext: this, EditContact.class);
            i2.putExtra( name: "edit", value: false);
            startActivity(i2);
            break;
        case R.id.btn_menu_3:
            Intent ii = new Intent( packageContext: this, ListContact.class);
            ii.putExtra( name: "edit", value: true);
            startActivity(ii);
            break;
        case R.id.btn_menu_4:
            contactDB_dao.videLaBase();
            break;
    }
}
```

Par la suite, lorsque l'utilisateur choisit un des contacts, une nouvelle activité est affichée, initialisée par le nom duContact à modifier. L'enchaînement des activités peut être:



Il faudra donc passer le `Contact` d'une activité à une autre. Pour cela on utilisera le code (peut être à adapter) :

```
Contact ctAModifier = arContacts.get(position);
Intent i = new Intent(getApplicationContext(), ContactAModifierActivity.class);
Bundle b = new Bundle();
b.putSerializable(Constants.CONTACT, ctAModifier);
i.putExtras(b);
startActivity(i);
```

Pour récupérer ce `Contact` dans l'activité destinataire on peut écrire le code :

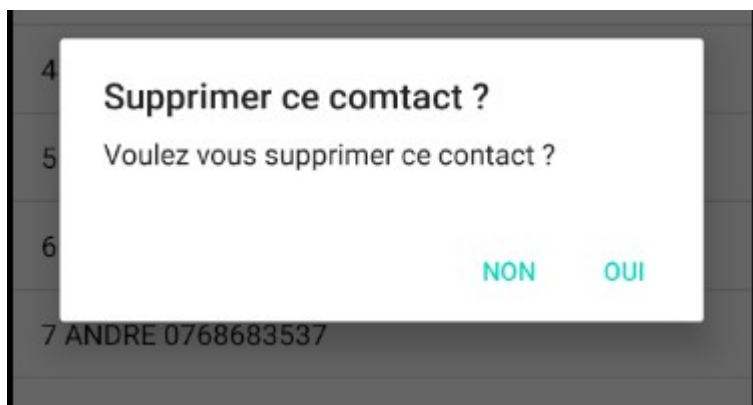
```
Bundle b = this getIntent().getExtras();
if (b != null) {
    ct = (Contact)(b.getSerializable(Constants.CONTACT));
}
```

10°) Enrichir l'activité qui présente tous les contacts dans une `ListView` pour pouvoir en détruire le contact choisi par l'utilisateur. On pourra construire de nouvelles activités ou enrichir celles déjà développées.

```
@Override
public boolean onItemClick(AdapterView<?> parent, View view, int position, long id) {
    Log.i( tag: "tag", msg: "ok");
    new AlertDialog.Builder( context: this)
        .setTitle("Supprimer ce contact ?")
        .setMessage("Voulez vous supprimer ce contact ?")
        .setPositiveButton( text: "oui", listener: null)
        .setNegativeButton( text: "non", listener: null)
        .show();

    return true;
}
```

SQLITE TD6	
Listes des contacts :	
1	Schonn 0652460461
2	ANDRE 0768683537
3	Mathieu 0777776441
4	Alexandre 4328123122
5	William 59292523
6	Samuel 4238432423
7	ANDRE 0768683537
8	ANDRE 0768683537
9	ANDRE 0768683537



En restant appuyé
sur un contact, on
peut choisir de le
supprimer ou non

Conclusion :

Ce TD était intéressant puisque il m'a permis d'utiliser les listes en java en lien avec les bases de données SQLite.

Le plus compliqué fût de comprendre comment marche la Base de données sous AndroidStudio étant donné qu'il s'agit d'une base de données local à l'application