TP N°6

Initiation aux Bases de données SQLite (1)

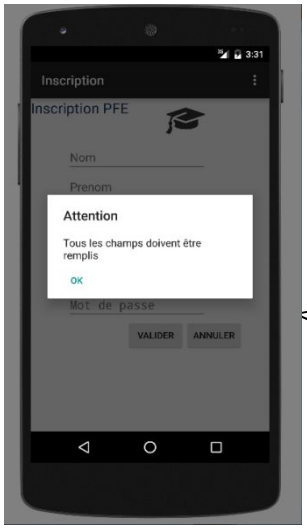
1. Interface graphique

Soit l'interface qui suit

Le premier écran affiché à l'étudiant est le suivant :

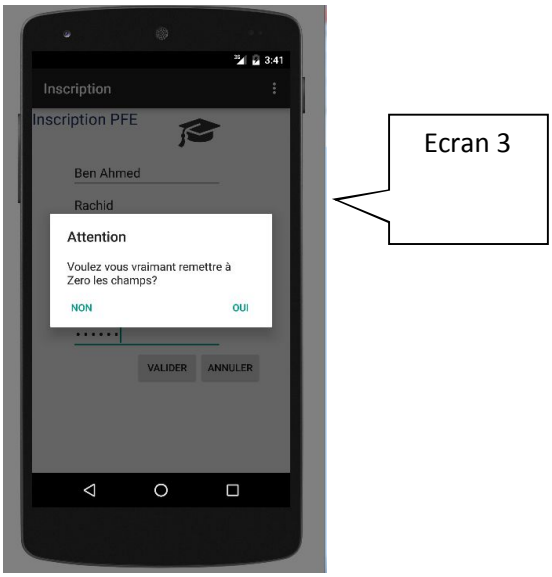
Ecran1

Si l'étudiant omet de remplir un champ, l'AlertDialog suivant s'affiche.



Ecran2

Si l'utilisateur décide d'annuler sa saisie, il peut appuyer sur le bouton « ANNULER » et l'écran suivant s'affiche



Q1. Créer l'interface d'inscription des étudiants.
Q2. Programmer les deux messages d'alertes visualisés par l'écran 2 et l'écran 3

Element	Id de l'objet dans le fichier id	Nom de l'objet dans la classe java
Ecran 1		
Le titre « Insription PFE »	Titre	
La zone de saisie « nom »	Nom	Nom
La zone de saisie « prénom »	Prenom	Prenom
La zone de saisie « Tel »	Phone	Tel
La zone de saisie « email »	Email	Email
La zone de saisie « login »	Login	Login
La zone de saisie « Mot de passe »	Mdp	MDP
Le bouton « VALIDER »	btnValider	
Le bouton « ANNULER »	btnAnnuler	

2. Création de la base de données

2.1 Création de la classe abstraite EtudiantBC

```
public class EtudiantBC {
    public EtudiantBC() {}
    public static abstract class EtudiantEntry implements BaseColumns
{
        public static final String TABLE_NAME = "etudiant";
        public static final String COLUMN_NAME_NOM = "nom";
        public static final String COLUMN_NAME_PRENOM = "prenom";
        public static final String COLUMN_NAME_PHONE = "phone";
        public static final String COLUMN_NAME_EMAIL = "email";
        public static final String COLUMN_NAME_LOGIN = "login";
        public static final String COLUMN_NAME_MDP = "mdp";
    }
```

```
        private static final String TEXT_TYPE = " TEXT";
        private static final String INT_TYPE = " INTEGER";
        private static final String COMMA_SEP = ",";
        public static final String SQL_CREATE_ENTRIES =
        "CREATE TABLE " + ……
    ……
    ……
    ……
    ……
    ……
    ……
    ……
    ……;

        public static final String SQL_DELETE_ENTRIES =
                "DROP TABLE IF EXISTS " + EtudiantEntry.TABLE_NAME;
    }
```

 Cette classe abstraite contient par défaut les champs suivants, on peut ne pas l'implémenter mais il est conseillé de travailler avec.

Constantes		
String	**_COUNT**	Nombre de lignes
String	**_ID**	La clé de la table

2.2 Création du DbHelper

```
public class EtudiantDBHelper extends SQLiteOpenHelper {
    public static final int DATABASE_VERSION = 1;
    public static final String DATABASE_NAME = "PFE.db";
    public EtudiantDBHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(EtudiantBC.SQL_CREATE_ENTRIES);
    }
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {
        db.execSQL(EtudiantBC.SQL_DELETE_ENTRIES);
        onCreate(db);
    }
    public void onDowngrade(SQLiteDatabase db, int oldVersion, int
newVersion) {
        onUpgrade(db, oldVersion, newVersion);
    }
```

```
}
```

3. **Insertion d'un Etudiant**

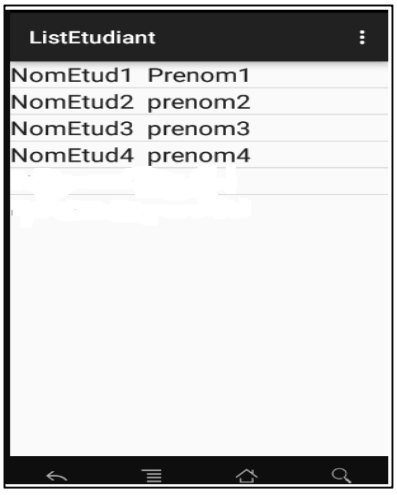
   Après avoir vérifié que les champs ne sont pas vides pendant le click sur le bouton **Valider,** l'étudiant est insérée dans la base par le code suivant :

```
ContentValues values = new ContentValues();
        values.put(EtudiantBC.EtudiantEntry.COLUMN_NAME_NOM,
Nom.getText().toString());
        values.put(……
……);
        values.put(……
……);
        values.put(……
……);
        values.put(……
……);
        values.put(……
……);
        EtudiantDBHelper mDbHelper = new
EtudiantDBHelper(getApplicationContext());
        SQLiteDatabase db = mDbHelper.getWritableDatabase();
        long newRowId;
        newRowId = db.insert(
                ……,
                ……,
                ……);

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

4. **Affichage de la liste d'étudiants**

   Une fois insérée, un écran contenant les noms des étudiants sera affiché comme suit



   Pour ce faire il faut :

a. Créer une nouvelle activité **ListEtudiant** dont l'interface contient initialement un **ListView (id=** `idlistetu`)

b. Créer un autre layout Ligne_Etudiant.xml représentant une ligne de la liste Etudiant
Il s'agit d'un TextView ayant comme id = nometud et un autre TextView (id=preetud)

c. Créer le DbHelper et la méthode getDbHelper comme suit

```
private EtudiantDBHelper dbHelper = null;
public EtudiantDBHelper getDbHelper() {
    if (dbHelper==null) {
        dbHelper = new EtudiantDBHelper(this);
    }
    return dbHelper;
}
```

d. Créer l'adaptateur à partir du curseur sql.

```
private SimpleCursorAdapter adapter = null;
public SimpleCursorAdapter getAdapter() {
    if (adapter==null) {
        SQLiteDatabase db = getDbHelper()
                .getReadableDatabase();
        Cursor c = db
                .rawQuery( "……" +
                        "……
"+……, null);
        adapter = new SimpleCursorAdapter(this,
                R.layout.ligne_etudiant,
                c,
new String[]
{EtudiantBC.EtudiantEntry.COLUMN_NAME_NOM,EtudiantBC.EtudiantEntry.COLUM
N_NAME_PRENOM },
new int[]{R.id.nometud, R.id.preetud},
,
                0);

    }
    return adapter;
}
```

e. Dans la méthode onCreate créer votre adaptateur et l'affecter à la liste Etudiants

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_list_etudiant);
    ListEtudiant=(ListView)findViewById(R.id.idlistetu);
    ListEtudiant.setAdapter(getAdapter());
}
```

f. Revenir à la première interface d'insertion pour programmer le passage vers la liste d'étudiants