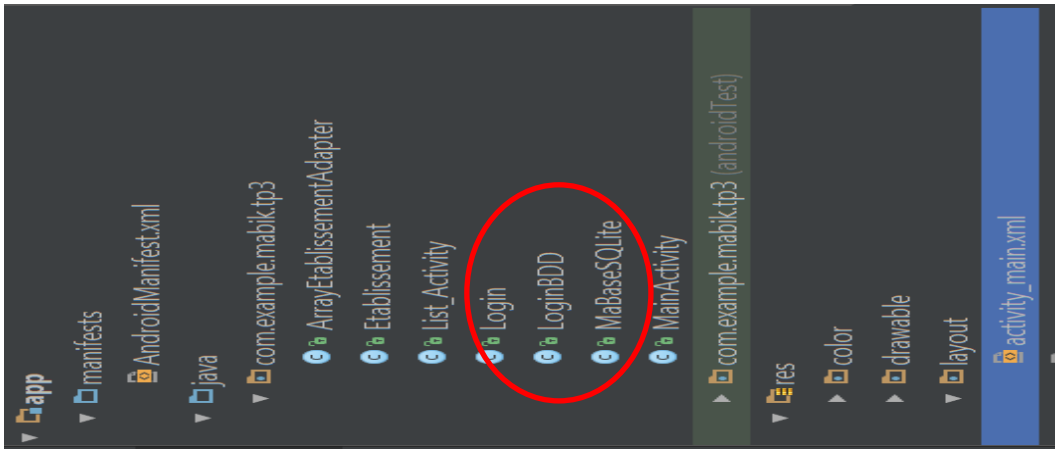
TP3

Afficher la ListView si le compte existe.

Création d'un nouveau compte s'il n'existe pas. Sinon on avertit l'utilisateur

L'objectif de ce Tp est la manipulation d'une simple base de données SQLITE pour gérer les login et les mots de passe via l'interface ci-dessus.

La structure et les consignes présentées ci-dessous vous guident à réaliser ce TP.



Etape 1 :

- Pour la gestion de la zone de texte cliquable associé à la création d'un nouveau compte (avec changement de couleur) :
  - Ajoutez dans le fichier layout principal, le TextView pour l'affichage de « Créer un nouveau compte » puis y ajoutez

```
android:textColor="@color/couleurs"
android:clickable="true"
android:selectAllOnFocus="true"
```

http://developer.android.com/reference/android/widget/TextView.html

- Ajoutez le dossier color (dans res) contenant le fichier couleurs.xml puis insérez dans ce dernier ce qui suit :

```
<selector xmlns:android="http://schemas.android.com/apk/res/android" >
     <item android:state_selected="true" android:color="#87E990" />
        <item android:state_focused="true" android:color="#FF8000" />
        <item android:state_pressed="true" android:color="#0000FF"  />
        <item android:color="#000000" />
</selector>
```

http://developer.android.com/guide/topics/resources/color-list-resource.html

Etape 2 :

- Construction de la class Login

```
public class Login {

      private int id;
      private String login;
      private String pass;

      public Login(){}

      public Login(String login, String pass){
            this.login = login;
            this.pass = pass;
      }
```

Complétez la classe avec les méthodes getID(), setID(int), getLogin(), setLogin(string), getPass(), setPass(string).

Etape 3 :

- Ajoutez une classe qui hérite SQLiteOpenHelper pour la création et la mise à jour de la BD.

```
public class MaBaseSQLite extends SQLiteOpenHelper {

    private static final String TABLE_LOGIN = "table_login";
    private static final String COL_ID = "ID";
    private static final String COL_login = "Login";
    private static final String COL_pass = "Pass";


    private static final String CREATE_BDD = "CREATE TABLE " + TABLE_LOGIN + " ("
    + COL_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " + COL_login + " TEXT NOT NULL, "
    + COL_pass + " TEXT NOT NULL);";

    public MaBaseSQLite(Context context, String name, CursorFactory factory, int version) {
        super(context, name, factory, version);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        //création de la table à partir de la requête écrite dans la variable CREATE_BDD
        db.execSQL(CREATE_BDD);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

        //déclenchée lors de chaque mise à jour de l'application par l'utilisateur
        //on supprime la table déjà existantes pour reconstruire une autre suivant le nouveau schéma
        db.execSQL("DROP TABLE " + TABLE_LOGIN + ";");
        onCreate(db);
    }

}
```

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

Etape 3 :

- Ajoutez la classe LoginBDD qui permet :
    - d'ouvrir la base de données en écriture et de la fermer

```
public class LoginBDD {

    private static final int VERSION_BDD = 1;
    private static final String NOM_BDD = "login.db";
    private static final String TABLE_LOGIN = "table_login";
    private static final String COL_ID = "ID";
    private static final int NUM_COL_ID = 0;
    private static final String COL_LOGIN = "login";
    private static final int NUM_COL_LOGIN = 1;
    private static final String COL_PASS = "Pass";
    private static final int NUM_COL_PASS = 2;
    private SQLiteDatabase bdd;
    private MaBaseSQLite maBaseSQLite;

    public LoginBDD(Context context){
        //création de la BDD
        maBaseSQLite = new MaBaseSQLite(context, NOM_BDD, null, VERSION_BDD);
                                }
        public void open(){
        //ouvrir la BDD en écriture
        bdd = maBaseSQLite.getWritableDatabase();
                                }
    public void close(){
        //fermeture de l'accès à la BDD
        bdd.close();
                                }
```

- d'insérer un compte (login et mot de passe) à la BD

```
public long insertlogin(Login login){
            //Création d'un objet ContentValues
            ContentValues values = new ContentValues();
            //on lui ajoute une valeur associé à une clé (qui est
le nom de la colonne dans laquelle on veut mettre la valeur)
            values.put(COL_LOGIN, login.getLogin());
            values.put(COL_PASS, login.getPass());

//insérer l'objet dans la BDD via le ContentValues

            return bdd.insert(TABLE_LOGIN, null, values);
                                        }
```

- de faire une recherche selon un login/mot_de_pass

```
public Login getLoginWithlogin(String log, String Pass){

//Récupérer dans un Cursor les valeurs correspondantes à un login)
Cursor c = bdd.query(TABLE_LOGIN, new String[] {COL_ID, COL_LOGIN,
COL_PASS}, COL_LOGIN  + " LIKE \"" + log + "\"" + " and " + COL_PASS  + "
LIKE \"" + Pass + "\"" , null, null, null, null);

return cursorToLogin(c);
}

//Cette méthode permet de convertir un cursor en un login
private Login cursorToLogin(Cursor c){
   //si aucun élément n'a été retourné par la requéte, on renvoie null
   if (c.getCount() == 0)
      return null;
      //Sinon on se place sur le premier élément
    c.moveToFirst();
   //instancier un login
    Login login = new Login();
   //on lui affecte toutes les infos à partir des infos contenues dans le
Cursor
   login.setId(c.getInt(NUM_COL_ID));
   login.setLogin(c.getString(NUM_COL_LOGIN));
   login.setPass(c.getString(NUM_COL_PASS));
   //On ferme le cursor
   c.close();
      //On retourne le login
   return login;
}
```

Etape 3 :

- Dans la méthode Myclick (TP1) ajoutez le traitement de recherche et d'ajout d'un nouveau compte dans la base de donnée (quand on clique sur « Créer un nouveau compte »)

- Dans l'activité principale, récupérez une instance de l'objet TextView (relatif à la création d'un nouveau compte) via findViewById(R.id…) puis gérez l'interaction (faire appel à la méthode Myclick)

- Modifiez la méthode Myclick pour vérifier l'existence du compte avant d'activer l'activité « List_Activity » (quand on clique sur le bouton « se connecter »)

Etape4 :

- Ajoutez d'autres méthodes (dans LoginBDD) pour supprimer par exemple un compte ou le modifier

Etape5 :

- Gérez les établissements à partir d'une Base de données en ajoutant/supprimant des établissements à partir du menu (TP2). Afficher la listView à partir de cette Base de données (comme source de données).

A vous..