Trimestre Hiver, 2016

Mohamed Lokbani

IFT1155 – Examen Final –

Inscrivez tout de suite : votre nom et le code permanent.

Nom : _____| Prénom(s) : _____|

Signature : _____| Login : _____|

Date : Mercredi 20 avril 2016

Durée : 3 heures (de 18h30 à 21h30)

Local : Z-220, Pavillon Claire McNicoll

Directives :

- Toute documentation est permise.
- Calculatrice **non** permise.

- Répondre directement sur le questionnaire.
- Les réponses **doivent être brèves, précises, claires et nettement présentées.**

1._____/20 (1.1 à 1.10)

2._____/20 (2.1, 2.2, 2.3, 2.4)

3._____/30 (3.1, 3.2, 3.3, 3.4)

4._____/30 (4.1, 4.2, 4.3, 4.4, 4.5)

Total : _____/100

Directives officielles

* Interdiction de toute communication verbale pendant l'examen.

* Interdiction de quitter la salle pendant la première heure.

* L'étudiant qui doit s'absenter après la première heure remettra sa carte d'étudiant au surveillant, l'absence ne devant pas dépasser 5 minutes. Un seul étudiant à la fois peut quitter la salle.

* Toute infraction relative à une fraude, un plagiat ou un copiage est signalée par le surveillant au directeur de département ou au professeur qui suspend l'évaluation.

F.A.S

Exercice 1 (20 points) Répondez par « vrai » ou « faux » en y incluant une très courte explication.

1.1 [VRAI | FAUX] La persistance des données des activités est gérée par l'objet « View ».

1.2 [VRAI | FAUX] L'utilisation de « SQLite » ne requiert aucune action d'installation ou d'administration de la part du développeur de l'application.

1.3 [VRAI | FAUX] La base de données d'une application est sauvegardée dans le répertoire « DATA/data/databases/Nom_Fichier ».

1.4 [VRAI | FAUX] Le chemin vers le répertoire principal (« DATA » dans le précédent exemple) est récupéré à l'aide de cette instruction : « DATA = Environment.getExternalStorageDirectory(); ».

1.5 [VRAI | FAUX] Le paquetage « android.database.sqlite » contient toutes les classes nécessaires pour travailler avec les bases de données.

1.6 [VRAI | FAUX] Un objet « SQLiteDatabase » est accessible à l'aide des deux méthodes « onCreate() » et « onUpgrade() ».

1.7 [VRAI | FAUX] Les méthodes « insert() », « update() » et « delete() » sont fournies par la classe « SQLiteOpenHelper ».

1.8 [VRAI | FAUX] Les « ContentValues » ont la même utilité (le même fonctionnent) que les Bundles.

1.9 [VRAI | FAUX] Les objets qui contiennent les résultats d'une recherche dans une base de données s'appellent « Widgets ».

1.10 [VRAI | FAUX] La méthode « execSQL() » est utilisée pour exécuter une requête « SQL » qui appelle une réponse, comme la sélection (« Select »).

Exercice 2 (20 points) Expliquez ce qui suit (ces définitions sont correctes) :

2.1 Android fournit plusieurs méthodes pour faire perdurer les données relatives à une application.

2.2 Avec Android, nous avons 3 façons d'accéder aux préférences qui semblent équivalentes : « getPreferences », « getSharedPreferences », « getDefaultSharedPreferences ».

2.3 Il est aussi possible de modifier des préférences par le code, par exemple si une action fait changer le paramétrage de l'application ou si l'on reçoit le paramétrage par le réseau.

2.4 Il est même possible de réagir à un changement de préférences en installant un écouteur sur celles-ci.

Exercice 3 (30 points)

3.1 Complétez le fichier « activity_splash.xml » pour obtenir la capture d'écran suivante:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="#ffffff" >
  <ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true" />
  <TextView
    android:layout_marginBottom="10dp"
    android:textSize="12dp"
    android:textColor="#454545"
    android:gravity="center_horizontal"
    android:layout_alignParentBottom="true" />
</RelativeLayout>
```

Il manque 2 éléments dans « ImageView » et 3 dans « TextView ».

3.2 Quelle est la tâche de cette classe

```
public class Splash extends Activity {
  private static int TEMPS = 3000;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_splash);
    new Handler().postDelayed(new Runnable() {
      @Override
      public void run() {
        Intent i = new Intent(Splash.this, MainActivity.class);
        startActivity(i);
        finish();
      }
    }, TEMPS);
  }
}
```

Pour cet exercice nous allons utiliser la classe « AsyncTask » pour récupérer le fichier « json » par internet. L'appel « HTTP » est transparent.

```
public class MainActivity extends ListActivity {
  // URL to get contacts JSON
  private static String url = "https://xyz";
  // JSON Node names
  private static final String TAG_ETUDIANTINFO = "etudiantsinfo";
  private static final String TAG_ID = "id";
  private static final String TAG_NOM = "nom";
  private static final String TAG_EMAIL = "courriel";
  private static final String TAG_ADRESSE = "adresse";
  private static final String TAG_GENRE = "genre";
  private static final String TAG_TEL = "tel";
  private static final String TAG_TEL_MOBILE = "mobile";
  private static final String TAG_TEL_MAISON = "maison";
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    new GetEtudiants().execute();
  }
  /* Plusieurs méthodes en rapport avec la classe Async task */
   private class GetEtudiants extends AsyncTask<Void, Void, Void> { }
}
```

3.3 Complétez la méthode « ParseJSON »

```
@Override
protected Void doInBackground(Void... arg0) {
  // Faire une demande du service web
  WebRequest webreq = new WebRequest();
  // Faire une requête url et obtenir une réponse
  String jsonStr = webreq.makeWebServiceCall(url, WebRequest.GET);
  etudiantListe = ParseJSON(jsonStr);
  return null;
}
```

Le contenu du fichier JSON est comme suit :

```
{
  "etudiantsinfo": [
    {
        "id": "100",
        "nom": "Adam",
        "courriel": "adam@example.com",
        "adresse": "House #33, ABC Lane, ABC CITY 06987",
        "genre" : "masculin",
        "tel": {
          "mobile": "+1 1234567890",
          "maison": "00 123456",
          "bureau": "00 321654"
        }
    },
    {
        "id": "102",
        "nom": "Mabell",
        "courriel": "mabell@example.com",
        "adresse": "House #33, ABC Lane, ABC CITY 06987",
        "genre" : "féminin",
        "tel": {
          "mobile": "+1 1234567890",
          "maison": "00 123456",
          "bureau": "00 123456"
        }
    }
  ]
}
```

3.4 Complétez la méthode « `onPostExecute` »

```
@Override
protected void onPostExecute(Void result) {
  super.onPostExecute(result);
  // Dismiss the progress dialog
  if (pDialog.isShowing())
    pDialog.dismiss();
  /**
   * Les données obtenues sont converties dans une ListView pour affichage
   * */
  ListAdapter adapter = ……..; // Ligne à compléter.
  setListAdapter(adapter);
}
```

Exercice 4 (30 points) Soit les éléments suivants :

```
private static final int DATABASE_VERSION = 1;
private static final String DATABASE_NOM = "MagasinsInfo";
private static final String TABLE_MAGASINS = "Magasins";
private static final String KEY_ID = "id";
private static final String KEY_NOM = "nom";
private static final String KEY_SH_ADR = "Magasin_adresse";
public DBHandler(Context context) {
  super(context, DATABASE_NOM, null, DATABASE_VERSION);
}
```

```
public class MainActivity extends AppCompatActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    DBHandler db = new DBHandler(this);
    Log.d("Ajout: ", "Insertions ..");
    db.addMagasin(new Magasin("Dockers", "475 Brannan St #330, San Francisco, CA 94107,
United States"));
    db.addMagasin(new Magasin("Dunkin Donuts", "White Plains, NY 10601"));
    db.addMagasin(new Magasin("Pizza Porlar", "North West Avenue, Boston , USA"));
    db.addMagasin(new Magasin("Town Bakers", "Beverly Hills, CA 90210, USA"));
    Log.d("Lecture: ", "Affichage de la liste des magasins ..");
    List<Magasin> Magasins = db.getAllMagasins();
    for (Magasin Magasin : Magasins) {
      String log = "Id: " + Magasin.getId() + " ,Nom: " + Magasin.getNom() + " ,Adresse: "
+ Magasin.getAdresse();
      Log.d("Magasin: : ", log);
    }
  }
}
```

4.1 Complétez la méthode « `onCreate(SQLiteDatabase db)` » qui permet de créer la table.

4.2 Complétez la méthode « `List<Magasin> getAllMagasins()` » qui permet d'afficher tous les magasins dans la base.

4.3 Complétez la méthode « `int getMagasinsNbre()` » qui permet d'obtenir le nombre de magasins dans la base.

4.4 Complétez la méthode « `int updateMagasin(Magasin Magasin)` » qui permet de mettre à jour les informations relatives à un magasin donné.

4.5 Après avoir exécuté 2 fois l'application, on remarque dans les logs un dédoublement de l'affichage. Nous avons 8 éléments au lieu de 4. Quelle est la variable qu'il faudra mettre à jour dans le programme afin d'éviter cette problématique. À l'aide de quelle méthode faudrait-il réaliser ce changement. Il n'est pas nécessaire d'écrire du code, mentionner le nom suffit.

Bon été!