Trimestre Hiver, 2017

Mohamed Lokbani

IFT1155 – Examen Final –

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

Nom : _____| Prénom(s) : _____|

Signature : _____| Login : _____|

Date : Mercredi 19 avril 2017

Durée : 3 heures (de 18h30 à 21h30)

Local : Z-317, 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)

2._____/20 (2.1, 2.2, 2.3, 2.4, 2.5)

3._____/20 (3.1)

4._____/40 (4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8)

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) trouvez la meilleure correspondance pour les descriptions ci-dessous en y associant un nombre de 1 à 11. Le nombre ne peut-être utilisé qu'une seule fois. Nous avons complété une description à titre d'exemple.

Un ____**programme** _(1) informatique est constitué d'instructions destinées au _____ (2).

On peut définir un _____(3) comme une instance d'un programme informatique qui est en cours d'exécution.

Beaucoup de gens pensent, à tort, qu'un _____(4) s'exécute dans un Thread secondaire.

Android est capable de détecter un ralentissement auquel cas il lancera une boîte de dialogue qui s'appelle _____(5).

Quand une activité est lancée, le système crée un _____(6) dans lequel s'exécutera l'application.

Un _____(7) représente une commande à exécuter, il est envoyé au _____(8) en utilisant un _____(9).

On fournit a une instance de la classe Thread, un _____(10) qui contient le _____(11) qui sera exécuté.

anr	
code	
handler	
message	
processeur	
processus	
programme	1
runnable	
thread	
thread principal	
service	

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

2.1 Expliquer l'utilité d'accompagner votre projet d'un rapport conséquent.

2.2 Les mobiles sont de plus en plus puissants, et leurs fonctionnalités toujours plus nombreuses sollicitent fortement l'autonomie de leur batterie au point qu'ils ne tiennent pas une journée.

2.3 Sous Android 5 (Lollipop) ou inférieur, à chaque installation d'application, le système vous demandera d'accepter ou non les permissions avant installation. Ce n'est plus le cas depuis Android 6 (Marshmallow).

2.4 Plusieures techniques sont mises à la disposition du développeur pour faire persister des données dans le temps.

2.5 Pour pouvoir utiliser Google Maps, il vous faudra demander l'autorisation pour accéder aux services sur internet. Ne pas confondre avec les permissions dans le fichier « Manifest.xml ».

Exercice 3 (20 points) cet exercice est en rapport avec les fonctionnalités d'une base de données. Le code fourni est incomplet. On vous demande de compléter les 10 lignes manquantes clairement indiquées.

activity_database.xml

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
03    xmlns:tools="http://schemas.android.com/tools"
04    android:layout_width="match_parent"
05    android:layout_height="match_parent"
06    tools:context="ca.umontreal.iro.ift1155.exo03.DatabaseActivity">
07    <TextView
08       android:id="@+id/produitID"
09       android:layout_width="wrap_content"
10       android:layout_height="wrap_content"
11       android:layout_alignParentTop="true"
12       android:layout_marginTop="36dp"
13       android:layout_toRightOf="@+id/button1"
14       android:text="@string/produit_id" />
15    <EditText
16       android:id="@+id/produitNom"
17       android:layout_width="wrap_content"
18       android:layout_height="wrap_content"
19       android:layout_alignRight="@+id/button2"
20       android:layout_below="@+id/produitID"
21       android:layout_marginTop="40dp"
22       ____________________________________________________________ #1
23    <EditText
24       android:id="@+id/produitQuantite"
25       android:layout_width="wrap_content"
26       android:layout_height="wrap_content"
27       android:layout_alignLeft="@+id/produitNom"
28       android:layout_below="@+id/produitNom"
29       android:layout_marginTop="50dp"
30       android:text="@string/produit_quantite"/>
31    <Button
32       ____________________________________________________________ #2
33       android:layout_width="wrap_content"
34       android:layout_height="wrap_content"
35       android:layout_alignParentBottom="true"
36       android:layout_alignParentLeft="true"
37       android:onClick="NouveauProduit"
38       android:text="Ajouter" />
39    <Button
40       android:id="@+id/button2"
41       android:layout_width="wrap_content"
42       android:layout_height="wrap_content"
43       android:layout_alignBaseline="@+id/button1"
44       android:layout_alignBottom="@+id/button1"
45       android:layout_centerHorizontal="true"
46       android:onClick="RechercheProduit"
47       android:text="Rechercher" />
48    <Button
49       android:id="@+id/button3"
50       android:onClick="RetirerProduit"
51       android:layout_width="wrap_content"
52       android:layout_height="wrap_content"
53       android:layout_alignBaseline="@+id/button2"
54       android:layout_alignBottom="@+id/button2"
55       android:layout_alignParentRight="true"
56       android:layout_marginRight="15dp"
57       android:text="Effacer" />
58 </RelativeLayout>
```

strings.xml

```
01 <?xml version="1.0" encoding="utf-8"?>
02    <resources>
03       <string name="app_name">Exo3_DatabaseActivity</string>
04       <string name="action_settings">Settings</string>
05       <string name="produit_nom">Produit Nom</string>
06       ____________________________________________________________ #3
07    <string name="produit_quantite">Produit Quantite</string>
08    </resources>
```

Produit.java

```
01  package ca.umontreal.iro.ift1155.exo03;
02  public class Produit {
03    private int _id;
04    private String _nomproduit;
05    private int _quantite;
06    public Produit() {
07    }
08    public Produit(int id, String nomproduit, int quantite) {
09      this._id = id;
10      this._nomproduit = nomproduit;
11      this._quantite = quantite;
12    }
13    public Produit(String nomproduit, int quantite) {
14      this._nomproduit = nomproduit;
15      this._quantite = quantite;
16    }
17    public void setID(int id) {
18      this._id = id;
19    }
20    public int getID() {
21      ____________________________________________________________
22    }
23    public void setNomProduit(String nomproduit) {
24      this._nomproduit = nomproduit;
25    }
26    public String getNomProduit() {
27      return this._nomproduit;
28    }
29    public void setQuantite(int quantite) {
30      this._quantite = quantite;
31    }
32    public int getQuantite() {
33      return this._quantite;
34    }
35  }
```

#4

DatabaseActivity.java

```
01  package ca.umontreal.iro.ift1155.exo03;
02  import android.os.Bundle;
03  import android.app.Activity;
04  import android.view.View;
05  import android.widget.EditText;
06  import android.widget.TextView;
07  public class DatabaseActivity extends Activity {
08      TextView idView;
09      EditText produitBox;
10      EditText quantiteBox;
11      @Override
12      protected void onCreate(Bundle savedInstanceState) {
13          super.onCreate(savedInstanceState);
14          setContentView(R.layout.activity_database);
15          ____________________________________________________________
16          produitBox = (EditText) findViewById(R.id.produitNom);
17          quantiteBox = (EditText) findViewById(R.id.produitQuantite);
18      }
19      public void NouveauProduit (View view) {
20          MyDBHandler dbHandler = new MyDBHandler(this, null, null, 1);
21          int quantite =
22                  Integer.parseInt(quantiteBox.getText().toString());
23          Produit produit =
24                  new Produit(produitBox.getText().toString(), quantite);
25          dbHandler.addProduct(produit);
26          produitBox.setText("");
27          quantiteBox.setText("");
28      }
29      public void RechercheProduit (View view) {
30          MyDBHandler dbHandler = new MyDBHandler(this, null, null, 1);
31          Produit produit =
32                  dbHandler.findProduct(produitBox.getText().toString());
33          if (produit != null) {
34              idView.setText(String.valueOf(produit.getID()));
35              quantiteBox.setText(String.valueOf(produit.getQuantite()));
36          } else {
37              ____________________________________________________________
38          }
39      }
40      public void RetirerProduit (View view) {
41          MyDBHandler dbHandler = new MyDBHandler(this, null, null, 1);
42          boolean result = dbHandler.deleteProduct(
43                  produitBox.getText().toString());
44          if (result) {
45              idView.setText("Enregistrement retire'");
46              produitBox.setText("");
47              quantiteBox.setText("");
48          }
49          else
50              idView.setText("Pas de match");
51      }
52  }
```

#5

#6

MyDBHandler.java

```
01 package ca.umontreal.iro.ift1155.exo03;
02 import android.content.ContentValues;
03 import android.content.Context;
04 import android.database.Cursor;
05 import android.database.sqlite.SQLiteDatabase;
06 import android.database.sqlite.SQLiteDatabase.CursorFactory;
07 import android.database.sqlite.SQLiteOpenHelper;
08 public class MyDBHandler extends SQLiteOpenHelper {
09    private static final int DATABASE_VERSION = 1;
10    private static final String DATABASE_NAME = "productDB.db";
11    private static final String TABLE_PRODUITS = "produits";
12    public static final String COLUMN_ID = "_id";
13    public static final String COLUMN_NOMPRODUIT = "nomproduit";
14 ____________________________________________________________ #7
15    public MyDBHandler(Context context, String name, CursorFactory factory, int version) {
16       super(context, DATABASE_NAME, factory, DATABASE_VERSION);
17    }
18    @Override
19    public void onCreate(SQLiteDatabase db) {
20       String CREATE_PRODUCTS_TABLE = "CREATE TABLE " +
21             TABLE_PRODUITS + "("
22             + COLUMN_ID + " INTEGER PRIMARY KEY," + COLUMN_NOMPRODUIT
23             + " TEXT," + COLUMN_QUANTITE + " INTEGER" + ")";
24       ____________________________________________________________ #8
25    }
26    @Override
27    public void onUpgrade(SQLiteDatabase db, int oldversion, int newversion) {
28       ____________________________________________________________ #9
29       onCreate(db);
30    }
31    public void ajouterProduit(Produit produit) {
32       ContentValues values = new ContentValues();
33       values.put(COLUMN_NOMPRODUIT, produit.getNomProduit());
34       values.put(COLUMN_QUANTITE, produit.getQuantite());
35       SQLiteDatabase db = this.getWritableDatabase();
36       db.insert(TABLE_PRODUITS, null, values);
37       db.close();
38    }
39    public Produit rechercherProduit(String nomproduit) {
40       String query = "Select * FROM " + TABLE_PRODUITS + " WHERE " +
41             COLUMN_NOMPRODUIT + " = \"" + nomproduit + "\"";
42       SQLiteDatabase db = this.getWritableDatabase();
43       ____________________________________________________________ #10
44       Produit produit = new Produit();
45       if (cursor.moveToFirst()) {
46          cursor.moveToFirst();
47          produit.setID(Integer.parseInt(cursor.getString(0)));
48          produit.setNomProduit(cursor.getString(1));
49          produit.setQuantite(Integer.parseInt(cursor.getString(2)));
50          cursor.close();
51       } else {
52          produit = null;
53       }
54       db.close();
55       return produit;
56    }
57    public boolean effacerProduit(String nomproduit) {
58       boolean result = false;
59       String query = "Select * FROM " + TABLE_PRODUITS + " WHERE " +
60             COLUMN_NOMPRODUIT + " = \"" + nomproduit + "\"";
61       SQLiteDatabase db = this.getWritableDatabase();
62       Cursor cursor = db.rawQuery(query, null);
63       Produit produit = new Produit();
64       if (cursor.moveToFirst()) {
65          produit.setID(Integer.parseInt(cursor.getString(0)));
66          db.delete(TABLE_PRODUITS, COLUMN_ID + " = ?",
67                new String[]{String.valueOf(produit.getID())});
68          cursor.close();
69          result = true;
70       }
71       db.close();
72       return result;
73    }
74 }
```

Exercice 4 (40 points) nous allons développer une application qui sert à envoyer via l'internet une information du type Json vers un serveur et recevoir une réponse de ce serveur suite à cet envoi.

4.1 dessiner l'interface graphique à partir du fichier XML ci-dessous, en prenant soin d'inclure un bref résumé de votre démarche.

activity_main.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical" >
  <!-- #FF0000: rouge -->
  <!-- #FFF: blanc -->
  <TextView
    android:id="@+id/tvIsConnected"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:background="#FF0000"
    android:textColor="#FFF"
    android:textSize="18dp"
    android:layout_marginBottom="5dp"
    android:text="is connected?" />
  <RelativeLayout
    android:layout_width="wrap_content"
   android:layout_height="wrap_content">
    <TextView
      android:id="@+id/tvName"
       android:layout_width="50dp"
     android:layout_height="wrap_content"
     android:text="Name"
     android:layout_alignBaseline="@+id/etName"/>
    <EditText
      android:id="@+id/etName"
       android:layout_width="150dp"
     android:layout_height="wrap_content"
     android:layout_toRightOf="@+id/tvName"/>
    <TextView
      android:id="@+id/tvCountry"
       android:layout_width="50dp"
     android:layout_height="wrap_content"
     android:layout_below="@+id/tvName"
     android:text="Country"
     android:layout_alignBaseline="@+id/etCountry"/>
    <EditText
      android:id="@+id/etCountry"
       android:layout_width="150dp"
     android:layout_height="wrap_content"
     android:layout_toRightOf="@+id/tvCountry"
     android:layout_below="@+id/etName"/>
    <TextView
      android:id="@+id/tvTwitter"
       android:layout_width="50dp"
     android:layout_height="wrap_content"
     android:layout_below="@+id/tvCountry"
     android:text="Twitter"
     android:layout_alignBaseline="@+id/etTwitter"/>
    <EditText
      android:id="@+id/etTwitter"
       android:layout_width="150dp"
     android:layout_height="wrap_content"
     android:layout_toRightOf="@+id/tvTwitter"
     android:layout_below="@+id/etCountry"/>
  </RelativeLayout>
  <Button
    android:id="@+id/btnPost"
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:text="ENVOI"/>
</LinearLayout>
```

strings.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">POST JSON</string>
    <string name="action_settings">Settings</string>
    <string name="hello_world">Hello world!</string>
</resources>
```

Person.java

```
package ca.umontreal.iro.ift1155.exo4.vo;
public class Person {
 private String name;
 private String country;
 private String twitter;

 public String getName() {
  return name;
 }
 public void setName(String name) {
  this.name = name;
 }
 public String getCountry() {
  return country;
 }
 public void setCountry(String country) {
  this.country = country;
 }
 public String getTwitter() {
  return twitter;
 }
 public void setTwitter(String twitter) {
  this.twitter = twitter;
 }
 @Override
 public String toString() {
  return "Person [name=" + name + ", country=" + country + ", twitter="
    + twitter + "]";
 }
}
```

Squelette du fichier MainActivity.java

```
package ca.umontreal.iro.ift1155.exo4;
import ……………

public class MainActivity extends Activity implements OnClickListener {

   TextView tvIsConnected;
   EditText etName,etCountry,etTwitter;
   Button btnPost;
   Person person;


  @Override
  protected void onCreate(Bundle savedInstanceState) {
  }

  public static String ENVOI(String urlStr, Person person){
  }

  @Override
  public void onClick(View view) {
  }

  public boolean isConnected(){
  }

  static class CustomHandler extends Handler {
  }

  private Handler messageHandler = new CustomHandler(this);

  private boolean validate(){
  }

  private static String convertInputStreamToString(InputStream inputStream) throws IOException{
  }
}
```

4.2 compléter la méthode « isConnected ». Cette méthode vérifie que nous avons accès au réseau (internet). Elle retourne « false » dans le cas contraire.

```
public boolean isConnected(){









}
```

4.3 compléter la méthode « validate ». Cette méthode valide que les éléments « etName », « etCountry » et « etTwitter » contiennent une chaîne de caractères. Elle retourne « false » si une des 3 chaînes est vide (« "" ».

```
private boolean validate(){









}
```

4.4 compléter la classe « CustomHandler ». Cette classe va gérer un handler. C'est une classe simpliste. Pour cet exercice, elle ne gère aucun message.

```
static class CustomHandler extends Handler {









}
```

4.5 compléter la méthode « convertInputStreamToString ». Cette méthode va convertir un flux « inputStream » lu en entrée vers une chaîne de caractères du type « String ». Le flux sera lu tant qu'il y a quelque chose à lire.

```
private static String convertInputStreamToString(InputStream inputStream) throws IOException{
```

```
}
```

4.6 compléter la méthode « onCreate ». Cette méthode est le point d'entrée de votre application. Elle va obtenir les références vers les différentes vues déclarées dans le fichier « activity_main.xml ». Par la suite, elle va faire appel à la méthode « isConnected » pour vérifier que nous avons accès au réseau. Si c'est le cas, elle va se servir de l'élément « `tvIsConnected` » pour afficher « `"You are conncted"` » en couleur verte. Dans le cas contraire, elle va afficher « `"You are NOT conncted"` » dans la couleur d'origine. Finalement, elle va juste définir un écouter sur le bouton « Envoi ».

```
protected void onCreate(Bundle savedInstanceState) {
```

```
}
```

4.7 compléter la méthode « onClick » associée au bouton « Post ». Cette méthode va s'assurer d'abord que nous avons bien cliqué sur le bouton « Post ». Elle valide à l'aide de la méthode « validate » que nous avons bien fourni des données, elle affiche un toast dans le cas contraire. Elle définit une instance de « Person » puis elle complète les champs nécessaires de cette instance. Elle démarre un thread dans lequel elle fait appel à la méthode « ENVOI » et envoie un message vide au handler. La signature de la méthode « ENVOI » est :

```
public static String ENVOI(String urlStr, Person person)
```

Le premier argument est la chaîne « `http://hmkcode.appspot.com/jsonservlet` », le second est l'instance de « Person » que vous avez déclarée. La méthode retourne un « String » dont le contenu sera affiché sur la sortie standard.

```
public void onClick(View view) {
```

```
}
```

4.8 compléter la méthode « ENVOI ». Cette méthode va :

-1. Définir une instance d'une URL dont la valeur est le 1er argument de la méthode « ENVOI ».
-2. Vérifier que cette URL est bien un « HTTP ».
-3 Définir les paramètres d'une connexion « HttpURLConnection ». À noter que nous allons utiliser ces configurations :
« `"Content-Type", "application/json"` », « `"Accept", "application/json"` » et « `"POST"` »
-4 Définir une instance de « JSONObject » et l'initialiser avec les paramètres du 2e argument de la méthode « ENVOI ».
-5 Exécuter la méthode « POST ».
-6 Obtenir une réponse du serveur et valider la réponse. Si la réponse est positive, on fait appel à la méthode « `convertInputStreamToString` ». Cette réponse est la valeur de retour de la méthode « ENVOI ». S'il n'y a pas de réponse, on renvoie le message « On a un bogue! ».

Bon été!