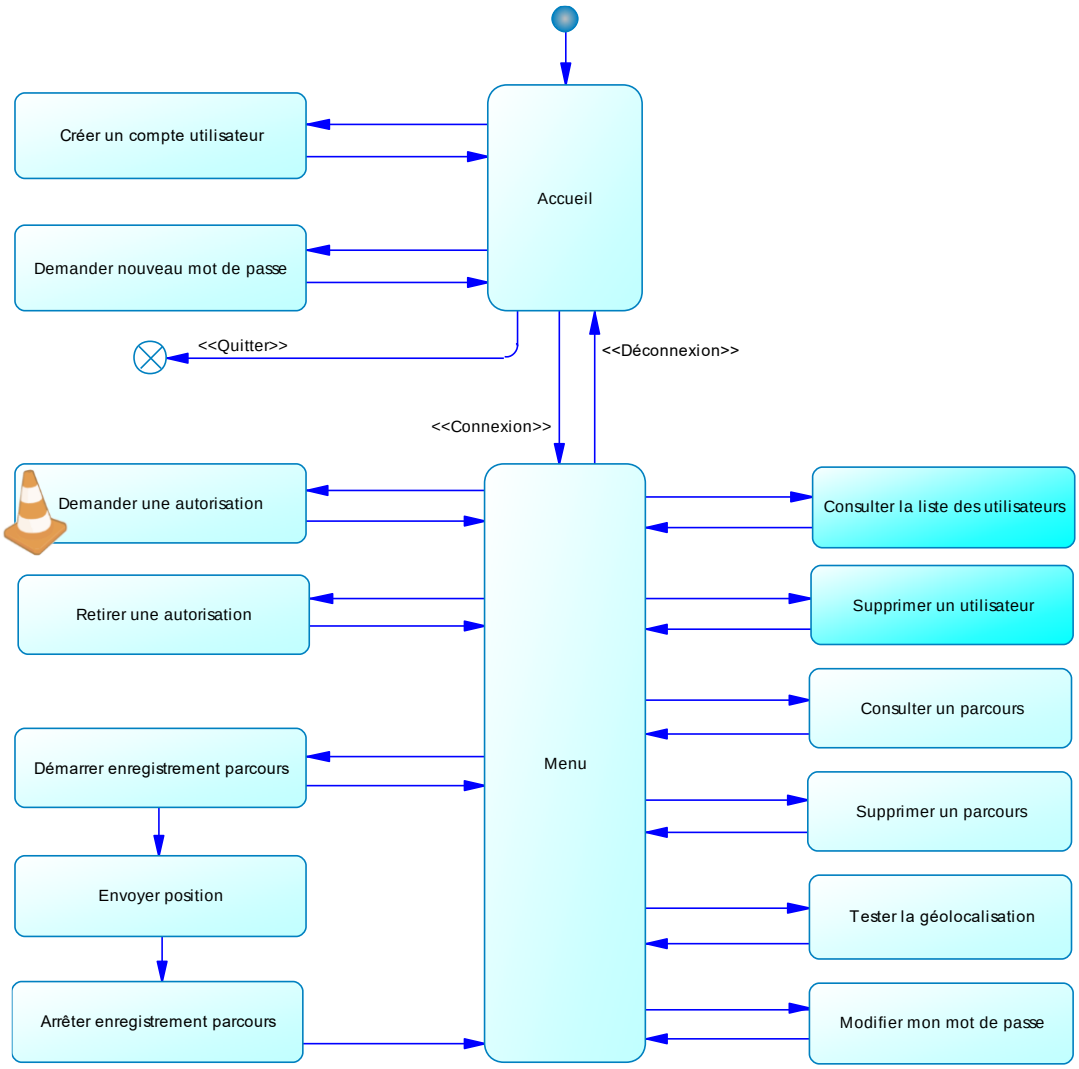
5- Développement de l'application mobile Android

5-5 DemanderAutorisation (demander une autorisation)

1- Situation de l'activité dans la structure de l'application
2- Modification du fichier strings.xml
3- Création de l'activité
4- Création de l'interface graphique
5- Modification de la programmation Java de MenuGeneral.java
6- Programmation Java de l'activité DemanderAutorisation.java

1- Situation de l'activité dans la structure de l'application



2- Modification du fichier strings.xml

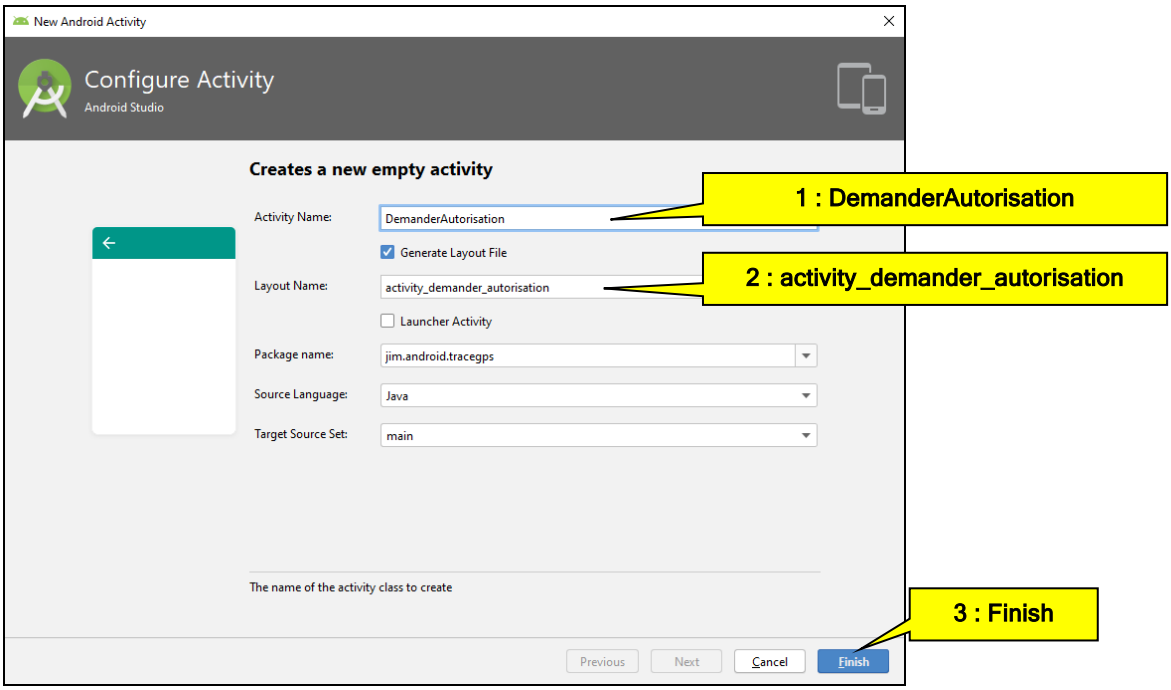
L'interface graphique à créer :

Dans le dossier **res/values**, complétez le fichier **strings.xml** avec le code suivant :

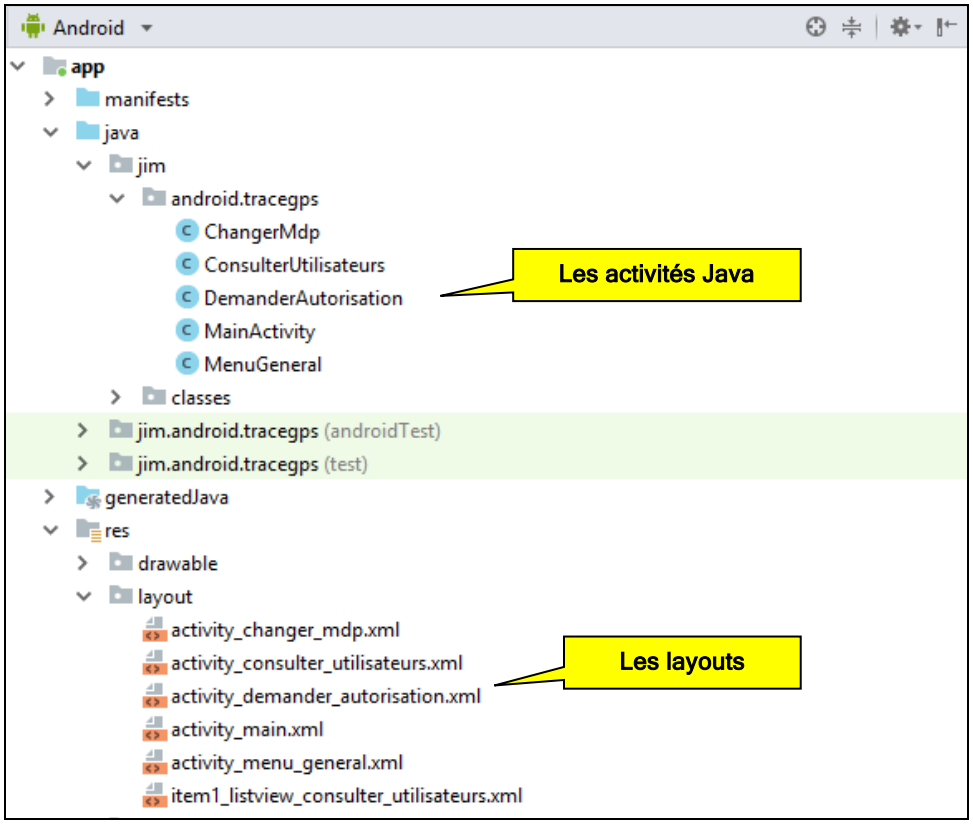
```
<!-- Les textes de la page de demande d'autorisation -->
<string name="demander_autorisation_titre1">Demander une autorisation</string>
<string name="demander_autorisation_bouton_retourner">Retourner au menu</string>
<string name="demander_autorisation_bouton_envoyer">Envoyer</string>
<string name="demander_autorisation_saisie_message">Votre message</string>
<string name="demander_autorisation_saisie_nom_prenom">Vos nom et prénom</string>
```

3- Création de l'activité

Créer une nouvelle activité en faisant un clic droit sur la racine **app** du projet et en choisissant la commande **New / Activity / Empty Activity** :



L'activité **DemanderAutorisation.java** et le layout **activity_demander_autorisation.xml** sont créés :



4- Création de l'interface graphique

A la création d'une nouvelle activité, l'interface comporte automatiquement un **ConstraintLayout** vide.

Comme d'habitude, nous allons commencer par remplacer le **ConstraintLayout** proposé par un **LinearLayout (vertical)** qui est beaucoup plus souple pour positionner les objets graphiques.

Le **ConstraintLayout** ne pouvant être ni modifié ni supprimé en mode **Design**, on va donc le modifier en mode **Text** :

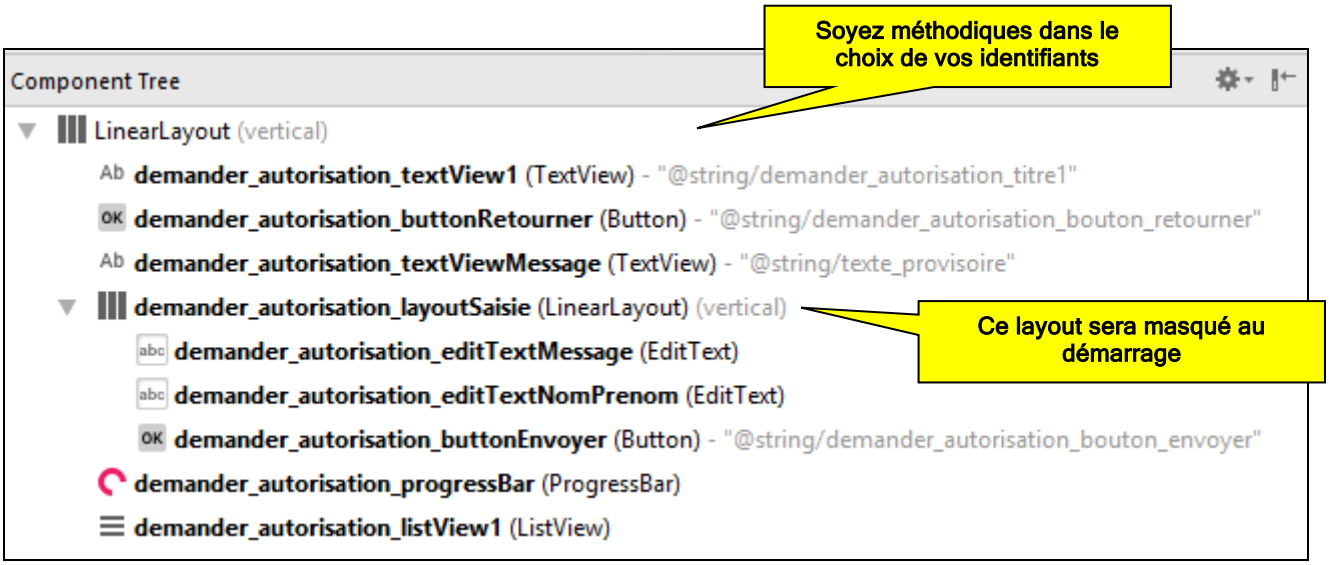
Remplacez ConstraintLayout par LinearLayout

```
<?xml version="1.0" encoding="utf-8"
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:padding="@dimen/tailleMarges"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="jim.android.tracegps.DemanderAutorisation">

</LinearLayout>
```

Supprimez cette ligne

Ajoutez ces 2 lignes

Revenez maintenant en mode **Design** et placez les différents composants en suivant la structure suivante et en utilisant bien sûr les chaines du fichier **strings.xml** :



Le code XML du layout :

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:padding="@dimen/tailleMarges"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="jim.android.tracegps.DemanderAutorisation">

    <TextView
        android:id="@+id/demander_autorisation_textView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingBottom="10dp"
        android:text="@string/demander_autorisation_titre1"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/demander_autorisation_buttonRetourner"
        style="@android:style/Widget.Button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/demander_autorisation_bouton_retourner"
        android:textSize="16sp" />

    <TextView
        android:id="@+id/demander_autorisation_textViewMessage"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="10dp"
        android:paddingBottom="10dp"
        android:text="@string/texte_provisoire"
        android:textSize="14sp" />

    <LinearLayout
        android:id="@+id/demander_autorisation_layoutSaisie"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <EditText
            android:id="@+id/demander_autorisation_editTextMessage"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="14sp"
        android:hint="@string/demander_autorisation_saisie_message" />

    <EditText
        android:id="@+id/demander_autorisation_editTextNomPrenom"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="14sp"
        android:hint="@string/demander_autorisation_saisie_nom_prenom" />

    <Button
        android:id="@+id/demander_autorisation_buttonEnvoyer"
        style="@android:style/Widget.Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:text="@string/demander_autorisation_bouton_envoyer"
        android:textSize="16sp" />

  </LinearLayout>

  <ProgressBar
    android:id="@+id/demander_autorisation_progressBar"
    style="?android:attr/progressBarStyle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal" />

  <ListView
    android:id="@+id/demander_autorisation_listView1"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />

</LinearLayout>
```

5- Modification de la programmation Java de MenuGeneral.java

Complétez l'écouteur d'événement associé à **buttonDemanderAutorisation** :

```
/** classe interne pour gérer le clic sur le bouton buttonDemanderAutorisation. */
private class buttonDemanderAutorisationClickListener implements View.OnClickListener{
    public void onClick(View v) {
        // crée une Intent pour lancer l'activité
        Intent uneIntent = new Intent(MenuGeneral.this, DemanderAutorisation.class);
        // passe nom, mdp et typeUtilisateur à l'Intent
        uneIntent.putExtra(EXTRA_PSEUDO, pseudo);
        uneIntent.putExtra(EXTRA_MDP, mdp);
        uneIntent.putExtra(EXTRA_TYPE_UTILISATEUR, typeUtilisateur);
        // démarre l'activité à partir de l'Intent
        startActivity(uneIntent);
    }
}
```

Testez cette étape sur un mobile réel **en vous connectant avec un des comptes utilisateurs**, et corrigez les erreurs si besoin.

Le bouton **Demander une autorisation** doit activer l'activité **DemanderAutorisation** :

6- Programmation Java de l'activité DemanderAutorisation.java

6-1 Déclarations diverses et initialisation des objets

Dans le fichier **DemanderAutorisation.java**, ajoutez le code indiqué en gras :

```
package jim.android.tracegps;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.ListView;
import android.widget.ProgressBar;
import android.widget.LinearLayout;
import android.widget.AdapterView;
import android.view.View;
import android.content.Intent;

public class DemanderAutorisation extends AppCompatActivity {

    // les objets du layout
    private TextView textViewMessage;           // le TextView pour afficher le message
    private EditText editTextMessage;           // l'EditText pour la saisie du message
    private EditText editTextNomPrenom;         // l'EditText pour la saisie du nom et prénom
    private Button buttonRetourner;             // le Button pour retourner au menu
    private Button buttonEnvoyer;               // le Button pour valider la demande
    private ProgressBar progressBar;            // le ProgressBar pour afficher le cercle de chargement
    private LinearLayout layoutSaisie;          // le layout contenant les 2 EditText et les 2 Button
    private ListView laListView;                // le ListView pour afficher les utilisateurs

    // le passage des données entre activités se fait au moyen des "extras" qui sont portés par les Intent.
    // un extra est une couple de clé/valeur
    // nous en utiliserons 2 ici, dont voici les 2 clés et les 2 variables associées :
    private final String EXTRA_PSEUDO = "pseudo";
    private final String EXTRA_MDP = "mdp";
    private String pseudo;
    private String mdp;

    private String texteMessage;            // pour la saisie du message
    private String nomPrenom;               // pour la saisie du nom et prénom

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_demander_autorisation);

        // récupération du nom, et du mot de passe passés par l'activité précédente
        Intent uneIntent = getIntent();
        pseudo = uneIntent.getStringExtra(EXTRA_PSEUDO);
        mdp = uneIntent.getStringExtra(EXTRA_MDP);

        // récupération des objets du layout grâce à leur ID
        textViewMessage = (TextView) findViewById(R.id.demander_autorisation_textViewMessage);
        editTextMessage = (EditText) findViewById(R.id.demander_autorisation_editTextMessage);
        editTextNomPrenom = (EditText) findViewById(R.id.demander_autorisation_editTextNomPrenom);
        buttonRetourner = (Button) findViewById(R.id.demander_autorisation_buttonRetourner);
        buttonEnvoyer = (Button) findViewById(R.id.demander_autorisation_buttonEnvoyer);
        progressBar = (ProgressBar) findViewById(R.id.demander_autorisation_progressBar);
        layoutSaisie = (LinearLayout) findViewById(R.id.demander_autorisation_layoutSaisie);
        laListView = (ListView) findViewById(R.id.demander_autorisation_listView1);
        // arrête le cercle de chargement
        progressBar.setVisibility(View.GONE);
```

```
        // association d'un écouteur d'évenement aux boutons
        buttonRetourner.setOnClickListener ( new buttonRetournerClickListener());
        buttonEnvoyer.setOnClickListener ( new buttonEnvoyerClickListener());

        // association d'un écouteur d'évenement à l'événement OnItemClic du ListView
        laListView.setOnItemClickListener ( new laListViewOnItemClickListener());
    }

    /** classe interne pour gérer le clic sur le bouton buttonRetourner. */
    private class buttonRetournerClickListener implements View.OnClickListener {
        public void onClick(View v) {
            finish();
        }
    }

    /** classe interne pour gérer le clic sur le bouton buttonEnvoyer. */
    private class buttonEnvoyerClickListener implements View.OnClickListener {
        public void onClick(View v) {
            // A COMPLETER PLUS TARD
        }
    }

    /** classe interne pour gérer le clic sur un item du ListView. */
    private class laListViewOnItemClickListener implements AdapterView.OnItemClickListener {
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            // A COMPLETER PLUS TARD
        }
    }
} // fin de l'activité
```

Testez l'application et le bon fonctionnement du bouton **Retourner au menu** :

6-2 Mise en place de 2 tâches asynchrones

L'affichage des utilisateurs nécessite d'appeler 2 services web :

- **GetTousLesUtilisateurs** : pour obtenir la liste des utilisateurs
- **GetLesUtilisateursQuiMautorisent** : pour obtenir la liste des utilisateurs qui m'autorisent déjà à suivre leurs parcours (afin de ne pas leur redemander une autorisation)

A la suite des **import** existants, ajoutez les **import** suivants :

```
import jim.classes.*;
import java.util.ArrayList;
import android.os.AsyncTask;
```

A la suite des déclarations existantes, ajoutez les déclarations suivantes :

```
private ArrayList<Utilisateur> lesUtilisateursQuiMautorisent;     // les utilisateurs qui m'autorisent
private ArrayList<Utilisateur> tousLesUtilisateurs;               // tous les utilisateurs
```

A la fin de l'activité, ajoutez les 2 tâches asynchrones **TacheGetLesUtilisateursQuiMautorisent** et **TacheGetTousLesUtilisateurs** (vous pouvez vous inspirer du document "***5-4 (0) Projet TraceGPS - Dév appli android - ConsulterUtilisateurs***").

```
// ------------------------------------------------------------------------------------------------------------------
// -------------------- tâche asynchrone pour PasserelleServicesWebXML.getLesUtilisateursQuiMautorisent ----------------------
// ------------------------------------------------------------------------------------------------------------------

private class TacheGetLesUtilisateursQuiMautorisent extends AsyncTask<ArrayList<Utilisateur>, Void, String> {

    La fonction onPreExecute doit démarrer l'affichage de l'objet progressBar.

    La fonction doInBackground doit appeler le service web GetLesUtilisateursQuiMautorisent en utilisant une des méthodes statiques de la classe PasserelleServicesWebXML et en lui passant les paramètres nécessaires. Cette méthode devra remplir la collection lesUtilisateursQuiMautorisent.

    La fonction onPostExecute doit tester le message retourné par le service web :

        • Si le message retourné par la méthode commence par le mot "Erreur", il faut arrêter l'affichage de l'objet progressBar et afficher dans l'objet textViewMessage le message retourné par la méthode

        • Sinon, il faut lancer l'exécution de la deuxième tâche asynchrone TacheGetTousLesUtilisateurs qui va être créée à l'étape suivante. Cette méthode devra remplir la collection tousLesUtilisateurs (à instancier avant)

}
```

```
// ------------------------------------------------------------------------------------------------------
// ------------------------- tâche asynchrone pour PasserelleServicesWebXML.getTousLesUtilisateurs -------------------------
// ------------------------------------------------------------------------------------------------------

private class TacheGetTousLesUtilisateurs extends AsyncTask<ArrayList<Utilisateur>, Void, String> {

    La fonction onPreExecute est inutile ici car l'affichage de l'objet progressBar est déjà démarré.

    La fonction doInBackground doit appeler le service web GetTousLesUtilisateurs en utilisant une des
    méthodes statiques de la classe PasserelleServicesWebXML et en lui passant les paramètres nécessaires.
    Cette méthode devra remplir la collection tousLesUtilisateurs.

    La fonction onPostExecute doit arrêter l'affichage de l'objet progressBar et tester le message retourné par le
    service web :

        • Si le message retourné par la méthode commence par le mot "Erreur", il faut afficher dans l'objet
          textViewMessage le message retourné par la méthode
        • Sinon, il faut exécuter la fonction afficherLesUtilisateurs dont le code provisoire est donné plus loin

}
```

A la fin de l'activité, ajoutez la fonction provisoire **afficherLesUtilisateurs** :

```
// afficher la liste des utilisateurs
public void afficherLesUtilisateurs() {
    // on affiche le nombre d'utilisateurs
    textViewMessage.setText(tousLesUtilisateurs.size() - 1 + " autres utilisateurs inscrits");
} // fin de la fonction afficherLesUtilisateurs
```

Complétez la fonction **onCreate** pour appeler la tâche **TacheGetLesUtilisateursQuiMautorisent** :

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_demander_autorisation);

    // récupération du nom, et du mot de passe passés par l'activité précédente
    Intent uneIntent = getIntent();
    pseudo = uneIntent.getStringExtra(EXTRA_PSEUDO);
    mdp = uneIntent.getStringExtra(EXTRA_MDP);

    // récupération des objets du layout grâce à leur ID
    textViewMessage = (TextView) findViewById(R.id.demander_autorisation_textViewMessage);
    editTextMessage = (EditText) findViewById(R.id.demander_autorisation_editTextMessage);
    editTextNomPrenom = (EditText) findViewById(R.id.demander_autorisation_editTextNomPrenom);
    buttonRetourner = (Button) findViewById(R.id.demander_autorisation_buttonRetourner);
    buttonEnvoyer = (Button) findViewById(R.id.demander_autorisation_buttonEnvoyer);
    progressBar = (ProgressBar) findViewById(R.id.demander_autorisation_progressBar);
    layoutSaisie = (LinearLayout) findViewById(R.id.demander_autorisation_layoutSaisie);
    laListView = (ListView) findViewById(R.id.demander_autorisation_listView1);
    // arrête le cercle de chargement
    progressBar.setVisibility(View.GONE);

    // association d'un écouteur d'évenement aux boutons
    buttonRetourner.setOnClickListener ( new buttonRetournerClickListener());
    buttonEnvoyer.setOnClickListener ( new buttonEnvoyerClickListener());

    // association d'un écouteur d'évenement à l'événement OnItemClic du ListView
    laListView.setOnItemClickListener ( new laListViewOnItemClickListener());

    Masquer le layout layoutSaisie.

    Effacer le message de l'objet textViewMessage.

    Instancier la collection lesUtilisateursQuiMautorisent.

    Lancer l'exécution de la tâche asynchrone TacheGetLesUtilisateursQuiMautorisent.
}
```

Testez l'application ; vous devez obtenir un affichage provisoire similaire à celui-ci :

6-3 Gestion de l'affichage des utilisateurs dans la ListView

A la suite des **import** existants, ajoutez l'**import** suivant :

```
import android.widget.ArrayAdapter;
```

A la suite des déclarations existantes, ajoutez la déclaration suivante :

```
private ArrayList<String> listeChaines;      // les libellés à placer dans le ListView
```

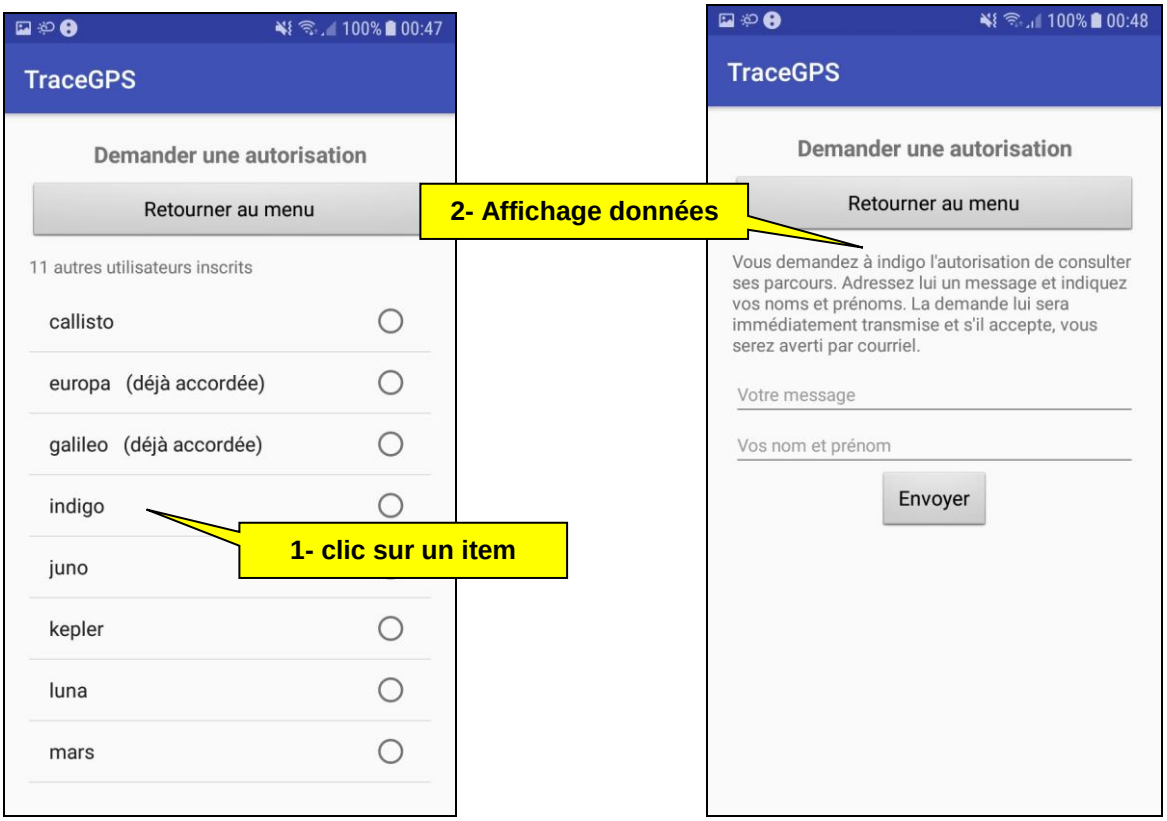
Complétez la fonction définitive **afficherLesUtilisateurs** :

```
// afficher la liste des utilisateurs
public void afficherLesUtilisateurs() {
    // on affiche le nombre d'utilisateurs
    textViewMessage.setText(tousLesUtilisateurs.size() - 1 + " autres utilisateurs inscrits");

    // vidage de la liste pour afficher les utilisateurs
    listeChaines = new ArrayList<String>();
    // parcours de l'ensemble des utilisateurs contenus dans tousLesUtilisateurs
    for (int i = 0; i < tousLesUtilisateurs.size(); i++)
    {  Utilisateur unUtilisateur = tousLesUtilisateurs.get(i);
       // ajout des utilisateurs à la liste (sauf le demandeur bien sûr)
       if ( ! unUtilisateur.getPseudo().equals(pseudo))
       {  String texte = unUtilisateur.getPseudo();
          // chercher si cet utilisateur a déjà accordé son autorisation
          for (Utilisateur unUtilisateurQuiMautorise : lesUtilisateursQuiMautorisent)
          {   if (unUtilisateur.getPseudo().equals(unUtilisateurQuiMautorise.getPseudo()))
              {   texte = unUtilisateur.getPseudo() + "   (déjà accordée)";
              }
          }
          listeChaines.add(texte);
       }
    }
    // le constructeur ArrayAdapter reçoit le contexte, le style des items, et les données à afficher :
    ArrayAdapter<String> monAdapter = new ArrayAdapter<String>(
          getBaseContext(),
          android.R.layout.simple_list_item_single_choice,
          listeChaines);
    // affichage de la liste
    laListView.setAdapter(monAdapter);
} // fin de la fonction afficherLesUtilisateurs
```

Testez l'application ; vous devez obtenir un affichage définitif similaire à celui-ci :

6-4 Gestion du clic sur un item du ListView

On va maintenant gérer le clic sur un item du ListView en masquant la ListView et en affichant le layout **layoutSaisie** :



A la suite des déclarations existantes, ajoutez la déclaration suivante :

```
private String pseudoDestinataire;          // le pseudo du destinataire choisi
```

Complétez l'écouteur d'événement **laListViewOnItemClickListener** :

```
/** classe interne pour gérer le clic sur un item du ListView. */
private class laListViewOnItemClickListener implements AdapterView.OnItemClickListener {
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        // recherche du pseudo choisi à partir de la position de l'item choisi
        pseudoDestinataire = listeChaines.get(position);
        // on ne traite pas le choix d'un utilisateur qui a déjà accordé son autorisation
        if ( ! pseudoDestinataire.endsWith("(déjà accordée)"))
        {
            laListView.setVisibility(View.GONE);
            layoutSaisie.setVisibility(View.VISIBLE);
            String msg = "Vous demandez à " + pseudoDestinataire + " l'autorisation de consulter ses parcours.";
            msg += " Adressez lui un message et indiquez vos noms et prénoms.";
            msg += " La demande lui sera immédiatement transmise et s'il accepte, vous serez averti par courriel.";
            textViewMessage.setText(msg);
        }
    }
}
```

position indique le numéro de l'item ayant reçu le clic

Exécutez et testez.

6-5 Gestion du clic sur le bouton de validation

Cette action nécessite d'appeler le service web :

- **DemanderUneAutorisation** : pour demander une autorisation à un utilisateur

A la suite des **import** existants, ajoutez l'**import** suivant :

```
import android.widget.Toast;
```

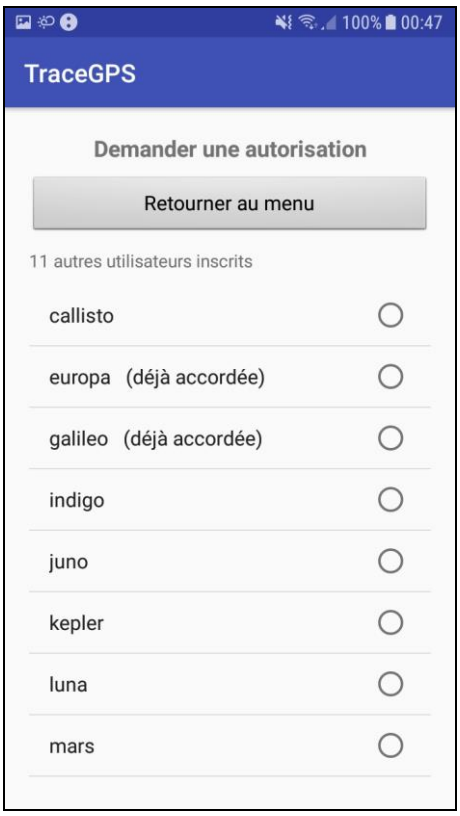
A la fin de l'activité, ajoutez la tâche asynchrone **TacheDemanderAutorisation** (vous pouvez vous inspirer du document "***5-3 (1) Projet TraceGPS - Dév appli android - ChangerMdp***") :

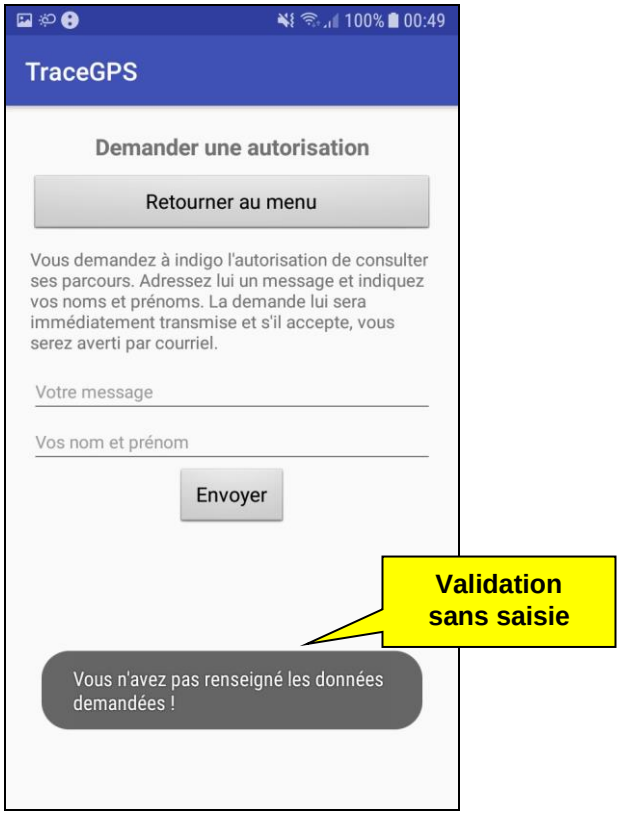
```
// ----------------------------------------------------------------------------------------------------------------------
// ---------- tâche asynchrone pour PasserelleServicesWebXML.demanderUneAutorisation ----------
// ----------------------------------------------------------------------------------------------------------------------

private class TacheDemanderAutorisation extends AsyncTask<Void, Void, String> {

    La fonction onPreExecute doit désactiver les boutons buttonRetourner et buttonEnvoyer et démarrer
    l'affichage de l'objet progressBar.

    La fonction doInBackground doit appeler le service web DemanderUneAutorisation en utilisant une des
    méthodes statiques de la classe PasserelleServicesWebXML et en lui passant les paramètres nécessaires.

    La fonction onPostExecute doit réactiver le bouton buttonRetourner, arrêter l'affichage de l'objet progressBar
    et afficher dans l'objet textViewMessage et dans un toast fugitif le message retourné par la méthode.

}
```

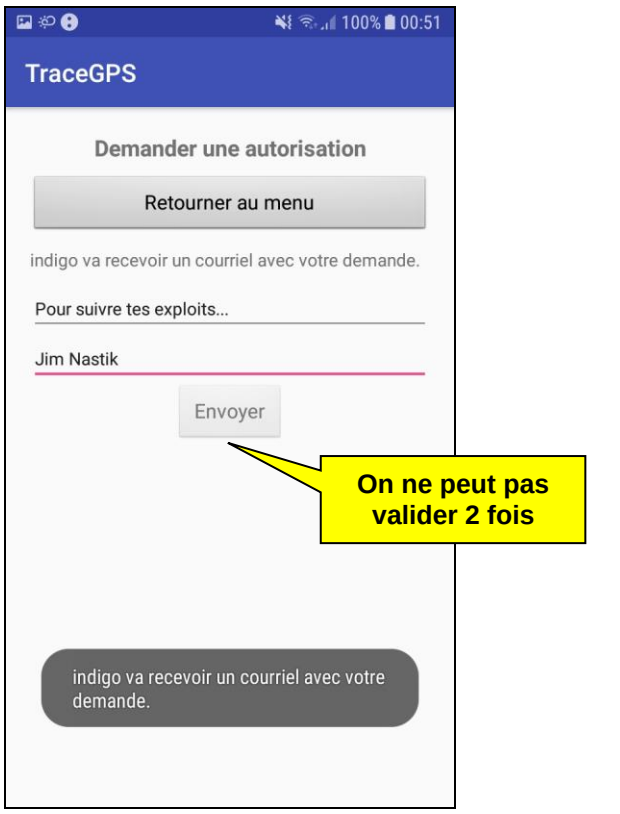
Complétez l'écouteur **buttonEnvoyerClickListener** pour appeler la tâche **TacheDemanderAutorisation** :

```
/** classe interne pour gérer le clic sur le bouton buttonEnvoyer. */
private class buttonEnvoyerClickListener implements View.OnClickListener {
    public void onClick(View v) {
        texteMessage = editTextMessage.getText().toString();
        nomPrenom = editTextNomPrenom.getText().toString();
        if (texteMessage.equals("") || nomPrenom.equals(""))
        {   String msg = "Vous n'avez pas renseigné les données demandées !";
            Toast.makeText(getApplicationContext(), msg, Toast.LENGTH_LONG).show();
        }
        else
        {   // appel du service web DemanderAutorisation avec une tâche asynchrone
            new TacheDemanderAutorisation().execute();
        }
    }
}
```

Testez l'application ; vous devez obtenir un dialogue similaire à celui-ci :







L'utilisateur visé par la demande doit recevoir un mail similaire à celui-ci :

Demande d'autorisation de la part d'un utilisateur du système TraceGPS Boîte de réception x

delasalle.sio.crib@gmail.com via 120-prod.mail-out.ovh.net
À moi

Cher ou chère indigo

Un utilisateur du système TraceGPS vous demande l'autorisation de suivre vos parcours.

Voici les données le concernant :

Son pseudo : helios
Son adresse mail : delasalle.sio.eleves@gmail.com
Son numéro de téléphone : 33.44.55.66.77
Son nom et prénom : Jim Nastik
Son message : Pour suivre tes exploits...

Pour accepter la demande, cliquez sur ce lien :
http://sio.lyceedelasalle.fr/tracegps/services/ValiderDemandeAutorisation.php?a=13e3668bbee30b004380052b086457b014504b3e&b=indigo&c=helios&d=1

Pour rejeter la demande, cliquez sur ce lien :
http://sio.lyceedelasalle.fr/tracegps/services/ValiderDemandeAutorisation.php?a=13e3668bbee30b004380052b086457b014504b3e&b=indigo&c=helios&d=0

Si l'utilisateur visé par la demande accepte, le demandeur doit recevoir un mail similaire à celui-ci :

Votre demande d'autorisation à un utilisateur du système TraceGPS Boîte de réception x

delasalle.sio.crib@gmail.com via 120-prod.mail-out.ovh.net
À moi

Cher ou chère helios

Vous avez demandé à indigo l'autorisation de consulter ses parcours.
Votre demande a été acceptée.

Cordialement.
L'administrateur du système TraceGPS

Le demandeur peut vérifier en retournant sur la page de demande d'autorisation :