les preuves.md

Les preuves prouvant les compétences demandées

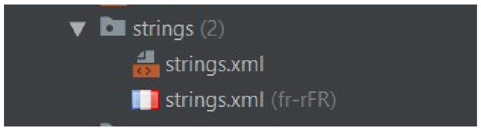
Je sais utiliser les Intent pour faire communiquer deux activités :

```
public void clickeventSettings(MenuItem item) {
    Intent intent = new Intent(this, SettingsActivity.class);
    startActivity(intent);
}
```

Je sais développer en utilisant le SDK le plus bas possible :

```
<uses-sdk
android:minSdkVersion = 15
android:targetSdkVersion = 28 />
```

Je sais distinguer mes ressources en utilisant les qualifier :



```
strings (2)
    strings.xml
    strings.xml (fr-rFR)
```

Je sais modifier le manifeste de l'application en fonction de mes besoins :

```
<activity
    android:name=".Activities.SettingsActivity"
    android:label="@string/title_activity_settings"
    android:parentActivityName=".MainActivity">
    <meta-data
        android:name="android.support.PARENT_ACTIVITY"
        android:value=".MainActivity" />
</activity>
```

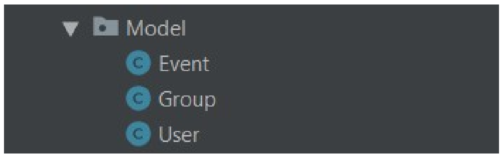
Je sais faire des vues xml en utilisant layouts et composants adéquats :

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:id="@+id/debugMessage"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/settings_title"
        android:textSize="@dimen/default_font_size"/>

</LinearLayout>
```

Je sais coder proprement mes activités, en m'assurant qu'elles ne font que relayer les évènements :

Nous avons mis toutes les classes métier dans le paquet Model pour être sûr que les activités ne fassent pas quelque chose qu'elles ne doivent pas faire.

Je sais coder une application en ayant un véritable métier :



Je sais parfaitement séparer vue et modèle :

Le modèle et la vue sont bien séparés car toutes les classes du modèle sont donc dans le paquet Model et pour les vues elles sont toutes dans le paquet layout.

Je maîtrise le cycle de vie de mon application :

```
//onCreate event
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_users);
}
//onStop event
@Override
protected void onSaveInstanceState(Bundle outState) {
    outState.putSerializable("users", (ArrayList<User>) listAllUsers);
    super.onSaveInstanceState(outState);
}
```

Je sais gérer les permissions dynamiques de mon application :

```
<!-- To auto-complete the email text field in the login form with the user's emails -->
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<uses-permission android:name="android.permission.READ_PROFILE" />
<uses-permission android:name="android.permission.READ_CONTACTS" />
```

Je sais gérer la persistance légère de mon application :

```
//saving the list of events
outState.putSerializable("events", (ArrayList<Event>) listEvents);

//get the list of events
listEvents = (ArrayList<Event>) savedInstanceState.getSerializable("events");
```

Je sais gérer la persistance profonde de mon application :

```
//save users in a file
fileOutputStream = openFileOutput("users", Context.MODE_PRIVATE);
outputStream = new ObjectOutputStream(fileOutputStream);
outputStream.writeObject(listAllUsers);
outputStream.close();
fileOutputStream.close();

//get users from a file
fileInputStream = openFileInput("users");
inputStream = new ObjectInputStream(fis);
listAllUsers = (ArrayList<User>) ois.readObject();
inputStream.close();
fileInputStream.close();
```

Je sais afficher une collection de données :

```
<!--Partie XML-->
<ListView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/listUsers"/>
```

```
//Partie code en java
private ListView listViewUsers;
listViewUsers = findViewById(R.id.listUsers);
// Personal ListView Adapter for Users
adapter = new UserAdapter(this, listAllUsers);
listViewUsers.setAdapter(adapter);
```

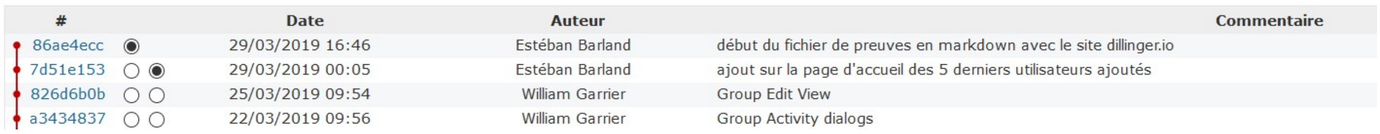
Je sais coder mon propre adaptateur :

```
public class GroupAdapter extends ArrayAdapter<Group> {
    private List<Group> listGroups;
    private Activity context;

    public GroupAdapter(Activity context, List<Group> listGroups) {
        super(context, R.layout.group_list_item, listGroups);
        this.context = context;
        this.listGroups = listGroups;
    }

    @NonNull
    @Override
    public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {
        LayoutInflater inflater = context.getLayoutInflater();
        View view = inflater.inflate(R.layout.group_list_item, null, false);

        TextView groupView = view.findViewById(R.id.nameGroupItemView);
        TextView descriptionGroup = view.findViewById(R.id.descriptionGroupItemView);

        Group group = listGroups.get(position);

        groupView.setText(group.getGroupName());
        descriptionGroup.setText(group.getDescription());

        return view;
    }
}
```

Je maîtrise l’usage des fragments :

```
public class UserInfoDialog extends DialogFragment {
    // User to display
    private User user;

    // Interface to send and receive data from the dialog to the activity
    public interface UserDeleteDialogListener {
        void onDeleteDialog(Bundle bundle);
    }

    @Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        // Get the user selected
        user = (User) getArguments().getSerializable("user");

        AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());

        LayoutInflater inflater = requireActivity().getLayoutInflater();
        builder.setView(inflater.inflate(R.layout.user_dialog, null))
                // Add action button
                .setNegativeButton(R.string.delete, new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        UserDeleteDialogListener listener = (UserDeleteDialogListener) getActivity();
                        listener.onDeleteDialog(getArguments());
                        dismiss();
                    }
                });
        return builder.create();
    }

    @Override
    public void onStart() {
        super.onStart();
        final Dialog dialog = getDialog();
        if(dialog != null) {
            // Get TextView Components of the dialog
            TextView firstname = dialog.findViewById(R.id.firstnameViewDialog);
            TextView surname = dialog.findViewById(R.id.surnameViewDialog);
            TextView email = dialog.findViewById(R.id.emailViewDialog);

            // Set the TextViews content by user's content
            firstname.setText(user.getFirstname());
            surname.setText(user.getSurname());
            email.setText(user.getEmail());
        }
    }
}
```

Je maîtrise l'utilisation de Git :



#		Date	Auteur	Commentaire
86ae4ecc	◉	29/03/2019 16:46	Estéban Barland	début du fichier de preuves en markdown avec le site dillinger.io
7d51e153	○ ◉	29/03/2019 00:05	Estéban Barland	ajout sur la page d'accueil des 5 derniers utilisateurs ajoutés
826d6b0b	○ ○	25/03/2019 09:54	William Garrier	Group Edit View
a3434837	○ ○	22/03/2019 09:56	William Garrier	Group Activity dialogs

Je sais utiliser l'accéléromètre :

```
public class ShakeEvent implements SensorEventListener {

    public interface OnShakeListener {
        void onShake();
    }

    private OnShakeListener listener;

    /***
     * Start the ShakeEvent
     * @param listener define the behaviour when a shake is detected
     */
    public void setOnShakeListener(OnShakeListener listener) {
        this.listener = listener;
    }

    /***
     * Stop the ShakeEvent
     */
    public void removeOnShakeListener() { sm.unregisterListener(this); }

    // Boolean to check if it is the first update of the values
    boolean firstUpdate = true;
    // Boolean to check if there is a shake
    boolean shake = false;
    // Minimum shake strength to detect the shake
    private static float SHAKE_THRESHOLD = 4f; // A TESTER SUR D'AUTRES TELEPHONES
    // Time limiter between every sensor update
    private static final long SHAKE_UPDATE_LIMITER = 1000l;
    // Last update time
    private long lastUpdate = System.currentTimeMillis();

    private float x, y, z;
    private float last_x, last_y, last_z;

    SensorManager sm;
    Sensor accelerometer;

    /***
     * ShakeEvent constructor
     * Build and register the sensor
     * @param sm SensorManager from the activity to initialize the sensor
     */
    public ShakeEvent(SensorManager sm){
        this.sm = sm;
        accelerometer = sm.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
        sm.registerListener(this, accelerometer, SensorManager.SENSOR_DELAY_NORMAL);
    }

    /***
     * Event that is called when a changed is detected on the sensor
     * @param event event type called by the sensor
     */
    @Override
    public void onSensorChanged(SensorEvent event) {
        long curTime = System.currentTimeMillis();
        if(curTime - lastUpdate > SHAKE_UPDATE_LIMITER) {
            updateValues(event.values[0], event.values[1], event.values[2]);
            if ((!shake) && isAccelerationChanged()) {
                shake = true;
            } else if ((shake) && isAccelerationChanged()) {
                listener.onShake();
            } else /*if ((shake) && !isAccelerationChanged())*/ {
                shake = false;
            }
            lastUpdate = curTime;
        }
    }

    /***
     * Verify if the acceleration has changed by comparing previous values and current values
     * @return true or false
     */
    private boolean isAccelerationChanged() {
        float dX = Math.abs(last_x - x);
        float dY = Math.abs(last_y - y);
        float dZ = Math.abs(last_z - z);
        return  (dX > SHAKE_THRESHOLD && dY > SHAKE_THRESHOLD) ||
                (dX > SHAKE_THRESHOLD && dZ > SHAKE_THRESHOLD) ||
                (dY > SHAKE_THRESHOLD && dZ > SHAKE_THRESHOLD);
    }

    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {

    }

    /***
     * Update the values
     * @param xNew new X value detected
     * @param yNew new Y value detected
     * @param zNew new Z value detected
```

```
         */
        public void updateValues(float xNew, float yNew, float zNew){
            if(firstUpdate){
                last_x = xNew;
                last_y = yNew;
                last_z = zNew;
                firstUpdate = false;
            }
            else {
                last_x = x;
                last_y = y;
                last_z = z;
            }
            x = xNew;
            y = yNew;
            z = zNew;
        }
    }
```