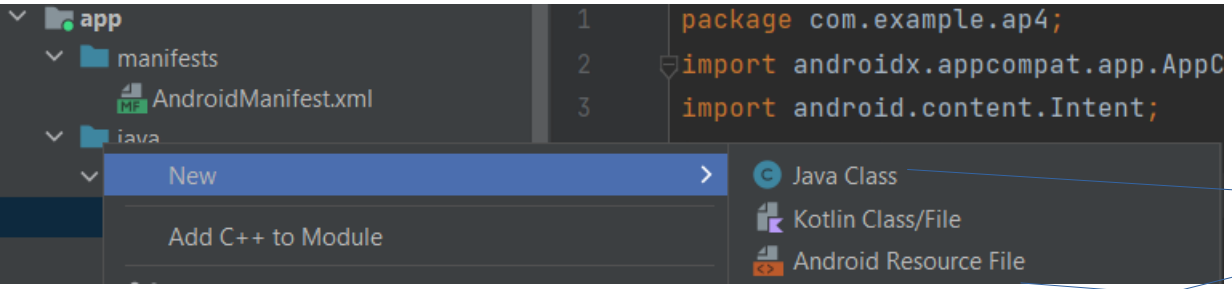
1. Créer une activité

A) Créer une classe java et un xml



Création d'une classe java

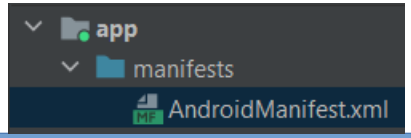
Création d'un xml

B) Associer la classe à son xml pour créer une activité

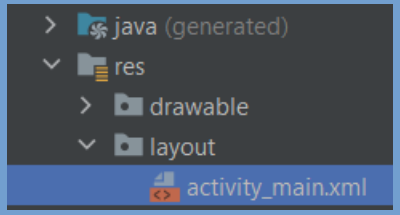
```java
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

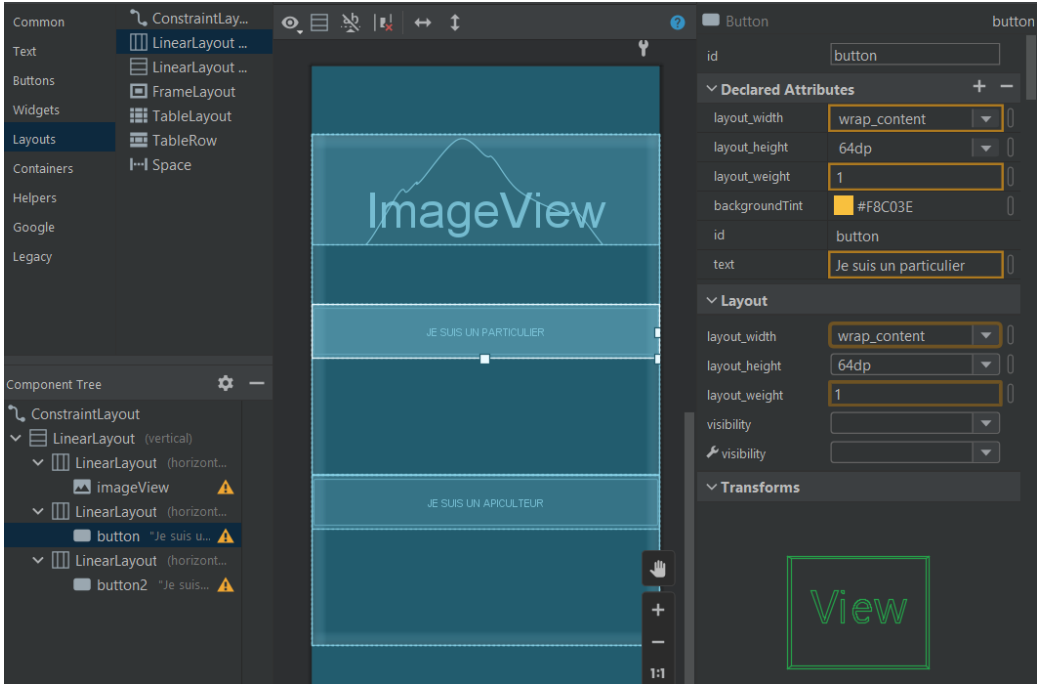
C) Le référencer dans le Manifest

```
<activity android:name=".particulier"></activity>
```



2. Gérer son xml





A) Structurer son xml

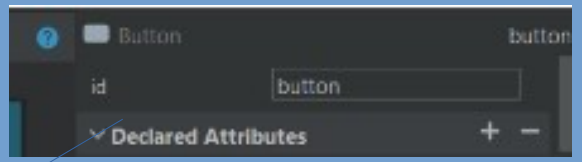
Utilisez LinearLayout Vertical, puis 2 à n fois LinearLayout Horizontal pour diviser l'écran en 2 à n parties.

Utilisez LinearLayout Horizontal, puis 2 à n fois LinearLayout pour diviser la partie en 2 à n fractions.

B) Ajouter des éléments

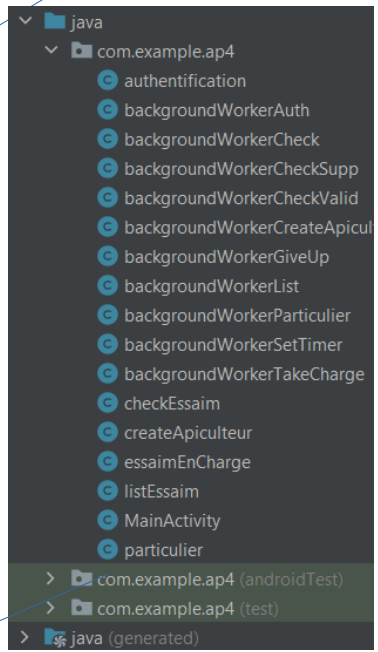
Ajoutez une image, un bouton, un textView ou un plainText pour permettre aux utilisateurs d'intéragir avec votre activité.

3. Utiliser un bouton et changer d'activité



A) Créer le constructeur et les import requis

```
import android.view.View;
import android.widget.Button;
import android.content.Intent;

public class MainActivity extends AppCompatActivity {
    private Button particulierBtn;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    particulierBtn = findViewById(R.id.button);
```



B) Créer la méthode bouton

```
particulierBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        // Ton code
    }
});
```

C) Code pour changer d'activité

```
Intent intent = new Intent(MainActivity.this, particulier.class);
startActivity(intent);
```

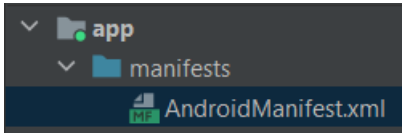
4. Récupérer la géolocalisation

```
import android.content.pm.PackageManager;
import android.location.Criteria;
import android.location.Location;
import android.location.LocationManager;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import android.Manifest;
```

A) Vérification des autorisations et récupération des données

```
if (ContextCompat.checkSelfPermission(particulier.this, Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED &&
        ContextCompat.checkSelfPermission(particulier.this, Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {

    ActivityCompat.requestPermissions(particulier.this,
            new String[]{Manifest.permission.ACCESS_FINE_LOCATION, Manifest.permission.ACCESS_COARSE_LOCATION},
            REQUEST_LOCATION_PERMISSION);
    return;
}
LocationManager locationManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
Criteria criteria = new Criteria();
criteria.setAccuracy(Criteria.ACCURACY_FINE);
String provider = locationManager.getBestProvider(criteria, true);
if (provider != null) {
    Location location = locationManager.getLastKnownLocation(provider);
    if (location != null) {
    // Ce que tu veux faire de ses données
```

B) Autoriser l'application à accéder à ses données




```
</application>
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

5. Autoriser la géolocalisation

```
import android.content.pm.PackageManager;
import android.location.Criteria;
import android.location.Location;
import android.location.LocationManager;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import android.Manifest;
```

```
private static final int REQUEST_LOCATION_PERMISSION = 123;
// Vérification des autorisations
if (ContextCompat.checkSelfPermission(this,
        Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
        ContextCompat.checkSelfPermission(this,
                Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {

    ActivityCompat.requestPermissions(this,
            new String[]{Manifest.permission.ACCESS_FINE_LOCATION, Manifest.permission.ACCESS_COARSE_LOCATION},
            REQUEST_LOCATION_PERMISSION);
} else {
    initLocation();
}

private void initLocation() {
    LocationManager locationManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
    Criteria criteria = new Criteria();
    criteria.setAccuracy(Criteria.ACCURACY_FINE);
    String provider = locationManager.getBestProvider(criteria, true);
    if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
            == PackageManager.PERMISSION_GRANTED) {
        if (provider != null) {
            Location location = locationManager.getLastKnownLocation(provider);
            if (location != null) {
                double latitude = location.getLatitude();
                double longitude = location.getLongitude();
            } else {
                Toast.makeText(particulier.this, "Impossible de récupérer votre position", Toast.LENGTH_SHORT).show();
            }
        } else {
            Toast.makeText(particulier.this, "Impossible de récupérer votre position", Toast.LENGTH_SHORT).show();
        }
    } else {
        Toast.makeText(particulier.this, "Autorisation refusée", Toast.LENGTH_SHORT).show();
    }
}
```

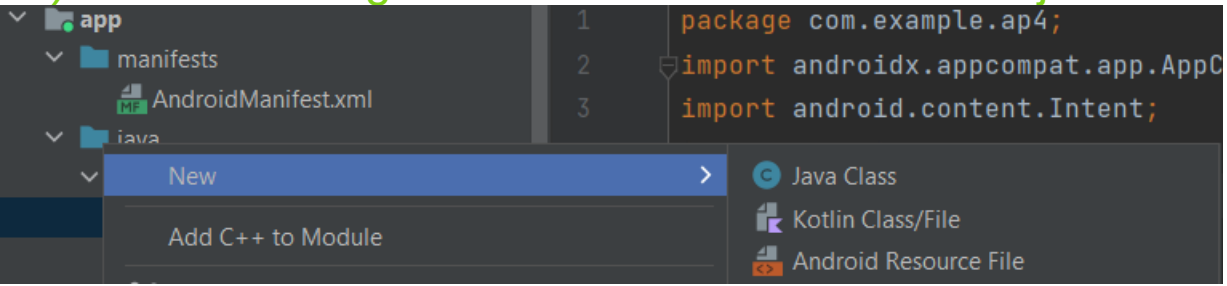
6. Créer une tâche en arrière plan d'une activité

A) Appeler un backgroundWorker concerné

```
String identifiant = identifiantEditText.getText().toString().trim();
String motdepasse = mdpEditText.getText().toString().trim();
String url = "http://adresseIP/authApiculteur.php";
String type = "authentification";
backgroundWorkerAuth backgroundWorker = new backgroundWorkerAuth(authentification.this);
backgroundWorker.execute(url, type, identifiant, motdepasse);
```

Identifiant et motdepasse sont des paramètres qu'on va utiliser en arrière plan.
L'url est l'adresse de localisation du fichier php hébergé qu'on va exécuter en arrière plan.
Le type est un string quelconque.
Authentification est le contexte de l'arrière plan.

B) Créer le backgroundWorker avec une classe java




```
public class backgroundWorkerAuth extends AsyncTask<String, Void, String> {
    Context context;
    AlertDialog alertDialog;
    backgroundWorkerAuth(Context context) {
        this.context = context;
    }
```

```
import android.app.AlertDialog;
import android.content.Context;
import android.os.AsyncTask;
```

7. Envoyer des données Android dans ton PHP

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.io.OutputStreamWriter;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLEncoder;
```

```
@Override
protected void onPreExecute() {
    alertDialog = new AlertDialog.Builder(context).create();
    return;
}
@Override
protected String doInBackground(String... params) {
    String login_url = params[0];
    String type = params[1];
    String identifiant = params[2];
    String motdepasse = params[3];
    try {
        // La requête POST HTTP
    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return null;
}
@Override
protected void onPostExecute(String result) {
// Ce que tu veux faire du résultat
}
@Override
protected void onProgressUpdate(Void... values) {
    super.onProgressUpdate(values);
}
```

```
URL url = new URL(login_url);
HttpURLConnection httpURLConnection = (HttpURLConnection) url.openConnection();
httpURLConnection.setRequestMethod("POST");
httpURLConnection.setDoOutput(true);
httpURLConnection.setDoInput(true);
OutputStream outputStream = httpURLConnection.getOutputStream();
BufferedWriter bufferedWriter = new BufferedWriter(new OutputStreamWriter(outputStream, "UTF-8"));
String post_data = "";
if (type.equals("authentification")) {
    post_data = URLEncoder.encode("identifiant", "UTF-8") + "=" + URLEncoder.encode(identifiant, "UTF-8") +
            "&"
            + URLEncoder.encode("motdepasse", "UTF-8") + "=" + URLEncoder.encode(motdepasse, "UTF-8");
}
bufferedWriter.write(post_data);
bufferedWriter.flush();
bufferedWriter.close();
outputStream.close();
InputStream inputStream = httpURLConnection.getInputStream();
BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(inputStream, "iso-8859-1"));
String result = "";
String line = "";
while ((line = bufferedReader.readLine()) != null) {
    result += line;
}
bufferedReader.close();
inputStream.close();
httpURLConnection.disconnect();
return result;
```

Type.equals doit correspondre au type du backgroundWorker.
Dans le post_data, les paramètres autres que url et type seront envoyés dans le php correspondant à l'url.

8. SQL pour les NULS

Type de Requête	Préparer la requête et l'exécuter

SELECT	$req = $bdd->query('SELECT * FROM table');
	while ($row = $req->fetch()) {
	  $results[] = $row;
	}

UPDATE	$req = $bdd->prepare('UPDATE table SET colonne1 = :nouvelleValeur WHERE condition'); $req->execute(array('nouvelleValeur' => $nouvelleValeur));
DELETE	$req = $bdd->prepare('DELETE FROM table WHERE condition'); $req->execute();

INSERT	$req = $bdd->prepare('INSERT INTO table (colonne1, colonne2) VALUES (:valeur1, :valeur2)');
	$req->execute(array(
	  'valeur1' => $valeur1,
	  'valeur2' => $valeur2
	));

JOIN	$req = $bdd->query('SELECT * FROM table1 INNER JOIN table2 ON table1.id = table2.table1_id');
	while ($row = $req->fetch()) {
	  $results[] = $row;
	}

Pour afficher les résultats de select

```
//Ta requête SELECT
$results = array();
//Ton while
foreach ($results as $row) {
    echo $row['colonne'] . "<br>";
}
```

9. Récupérer et renvoyer des données PHP vers Android avec le résultat d'une requête SQL

```php
<?php
// Récupération des données POST envoyées par l'application Android
$identifiant = $_POST['identifiant'];
$motdepasse = $_POST['motdepasse'];

// Connexion à la base de données MySQL
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "easybeebdd3";

$conn = mysqli_connect($servername, $username, $password, $dbname);
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// Requête SQL pour vérifier si l'apiculteur est enregistré dans la base de données
$sql = "SELECT * FROM apiculteur WHERE identifiant_apiculteur = '$identifiant' AND
motdepasse_apiculteur = '$motdepasse'";
$result = mysqli_query($conn, $sql);

// Si l'apiculteur est enregistré dans la base de données, on renvoie son identifiant
if (mysqli_num_rows($result) > 0) {
    $row = mysqli_fetch_assoc($result);
    $id_apiculteur = $row["id_apiculteur"];
    $essaimEnCharge = $row["essaimEnCharge"];
    $nom_apiculteur = $row["nom_apiculteur"];
    $prenom_apiculteur = $row["prenom_apiculteur"];
    $response = array('id_apiculteur' => $id_apiculteur, 'essaimEnCharge' => $essaimEnCharge,
'nom_apiculteur' => $nom_apiculteur, 'prenom_apiculteur' => $prenom_apiculteur);
    echo json_encode($response);
} else {
    echo "Identifiant ou mot de passe incorrect";
}
mysqli_close($conn);
?>
```

```java
import org.json.JSONException;
import org.json.JSONObject;

protected void onPostExecute(String result) {
    String nom_apiculteur = "";
    String prenom_apiculteur = "";
    System.out.println("le result" + result);
    try {
        JSONObject obj = new JSONObject(result);
        id_apiculteur = obj.getString("id_apiculteur");
        essaimEnCharge = obj.getString("essaimEnCharge");
        nom_apiculteur = obj.getString("nom_apiculteur");
        prenom_apiculteur = obj.getString("prenom_apiculteur");
    }
    catch(JSONException e) {
        e.printStackTrace();
    }
    System.out.println("id_apiculteur received : " + id_apiculteur);
    System.out.println("essaumEnCharge received : " + essaimEnCharge);
    return;
}
```

10. AlertDialog, Logs et Messages Toast

A) Créer un AlertDialog

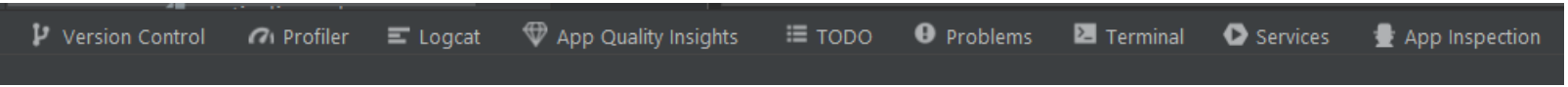
```
import android.app.AlertDialog;
import android.widget.Toast;
```

```
AlertDialog alertDialog;
alertDialog = new AlertDialog.Builder(context).create();
alertDialog.setTitle("Authentification reussie");
alertDialog.setMessage("Bonjour " + prenom_apiculteur + " " + nom_apiculteur);
alertDialog.show();
```

B) Créer et visualiser un message dans le Log

```
System.out.println("timer : " + timerText);
```

Le code sera visible dans Logcat :



C) Créer un message temporaire en bas de l'écran de l'activité

```
Toast.makeText(authentification.this, "Remplissez les deux zones de texte", Toast.LENGTH_SHORT).show();
```

On peut utiliser LENGHT_SHORT, LENGHT_LONG ou bien customiser avec ceci :

```
Toast.makeText(authentification.this, "Remplissez les deux zones de texte", 2000).show();
```

11. Méthode Handler et Exécuter après un onPostExecute

A) Créer une méthode Handler (une méthode qui se répète x millisecondes si tu veux)

```
import android.os.Handler;

private Handler handler = new Handler();
handler.postDelayed(new Runnable() {
    // Ton code
}, 1000); // Démarrer après une seconde (1000 millisecondes)
```

B) Exécuter un code après un onPostExecute en arrière plan

```
private OnPostExecuteListener onPostExecuteListener;

if (onPostExecuteListener != null) {
    onPostExecuteListener.onPostExecuteFinished();
}

public interface OnPostExecuteListener {
    void onPostExecuteFinished();
}

public void setOnPostExecuteListener(OnPostExecuteListener listener) {
    this.onPostExecuteListener = listener;
}

@Override
public void onPostExecuteFinished() {
    // Ton code
}
```

Écrire ce private dans le constructeur.

Écrire ce code à la toute fin du onPostExecute().

Ajouter ses méthodes au code de l'arrière plan.

Ajouter cette méthode dans l'activité.

12. Notifier les utilisateurs

```
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Intent;
import android.os.Build;
import androidx.core.app.NotificationCompat;
import androidx.core.app.NotificationManagerCompat;
import android.Manifest;
```

A) Créer un canal de discussion dans le onCreate

```
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
    CharSequence name = "Nom du canal";
    String description = "Description du canal";
    int importance = NotificationManager.IMPORTANCE_DEFAULT;
    NotificationChannel channel = new NotificationChannel("channelId", name, importance);
    channel.setDescription(description);
    NotificationManager notificationManager = getSystemService(NotificationManager.class);
    notificationManager.createNotificationChannel(channel);
}
```

B) Créer la notification dans onPostExecute

```
@Override
protected void onPostExecute(String result) {
    // ...
    NotificationCompat.Builder builder = new NotificationCompat.Builder(context, "channelId")
            .setSmallIcon(R.drawable.ic_launcher_foreground)
            .setContentTitle("Titre de la notification")
            .setContentText("Contenu de la notification")
            .setPriority(NotificationCompat.PRIORITY_DEFAULT);

    // Intent pour l'action de la notification
    Intent intent = new Intent(context, VotreActivite.class);
    intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
    PendingIntent pendingIntent = PendingIntent.getActivity(context, 0, intent,
PendingIntent.FLAG_UPDATE_CURRENT);
    builder.setContentIntent(pendingIntent);

    // Envoi de la notification
    NotificationManagerCompat notificationManager = NotificationManagerCompat.from(context);
    notificationManager.notify(0, builder.build());
    // ...
}
```

C) Créer l'autorisation de recevoir des notifications

```
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
    if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.ACCESS_FINE_LOCATION}, 101);
    }
}
```

Ajouter ceci dans le manifest :

```
<uses-permission android:name="android.permission.VIBRATE" />
```

13. Transférer la valeur d'une variable d'un arrière plan vers une activité autre que celle concernée et attendre n secondes

A) Créer méthode publique dans le constructeur

```
public static String essaimEnCharge = "null";
```

B) Créer un receptacle de la valeur d'une variable extérieure

```
private String idEssaim = backgroundWorkerAuth.essaimEnCharge;
```

Attention, puisque la variable de essaimEnCharge est reçue au moment d'un onPostExecute, il se peut que la valeur reçue soit nulle si on utilise trop vite le receptacle.
D'où l'importance de créer une méthode pour utiliser le receptacle après que onPostExecute soit pleinement exécutée ou d'attendre quelques secondes avant de l'utiliser.

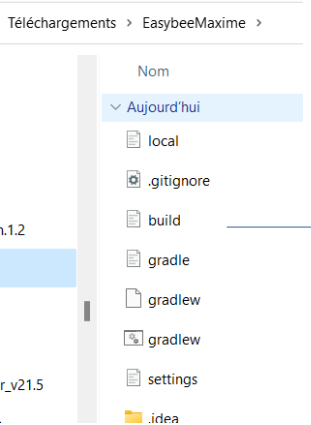
C) Utiliser un thread pour patienter n millisecondes

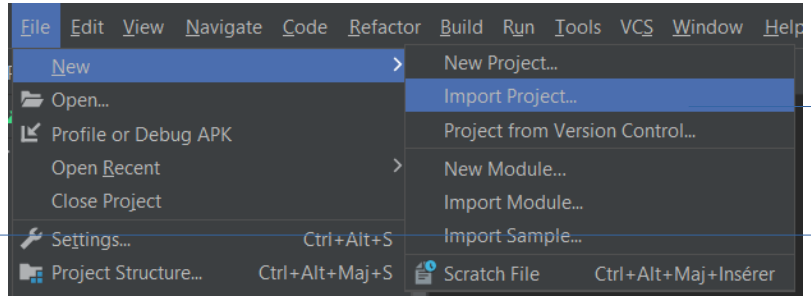
```
import java.lang.Thread;
try {
	// Attente de 2 secondes
	Thread.sleep(2000);
} catch (InterruptedException e) {
	e.printStackTrace();
}
```

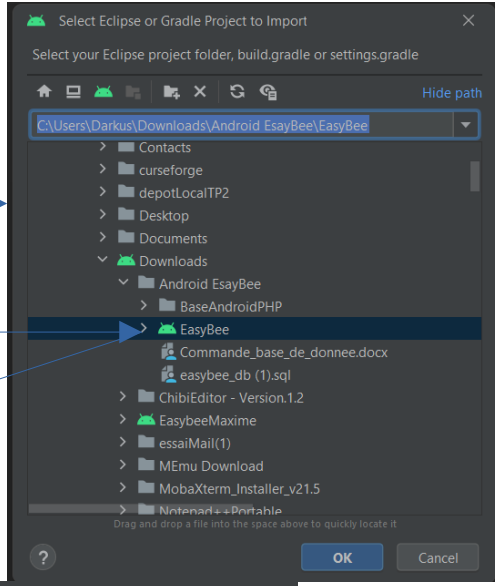
14. Importer un projet

A) La localisation de l'objet à importer

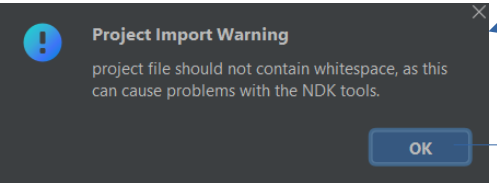
Le fichier à récupérer se situe à la racine de build.gradle

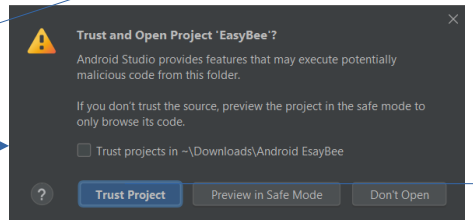


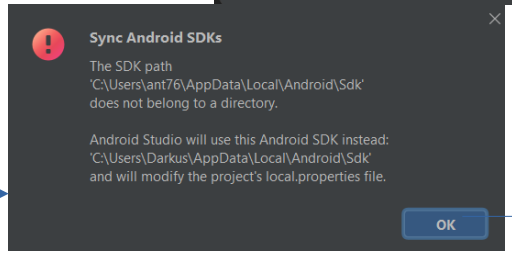




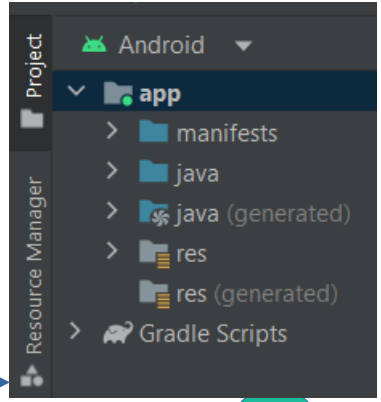
B) La récupération du projet sans safeMode





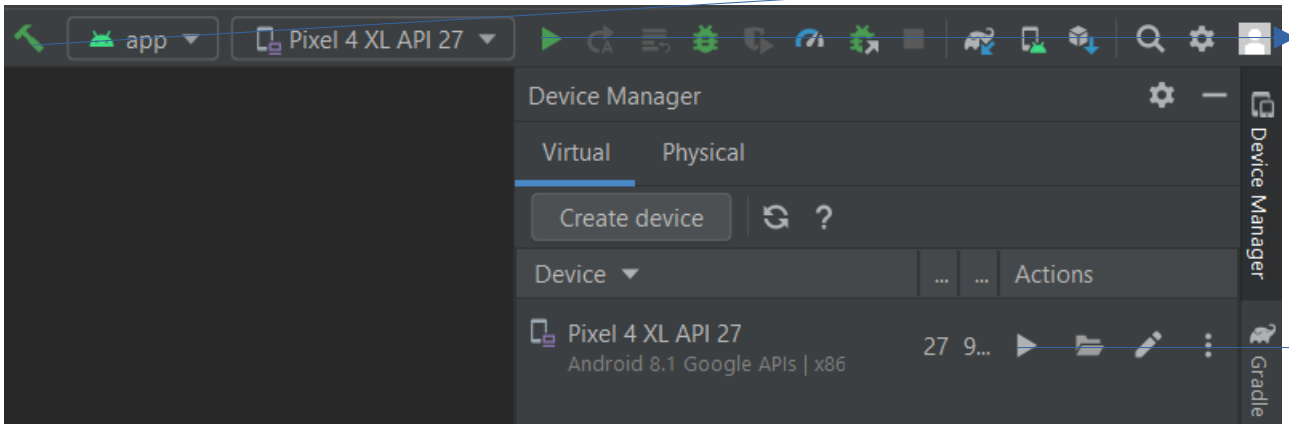


Projet récupéré



15. Compiler, exécuter son code et changer le nom de l'app

A) Les différentes actions

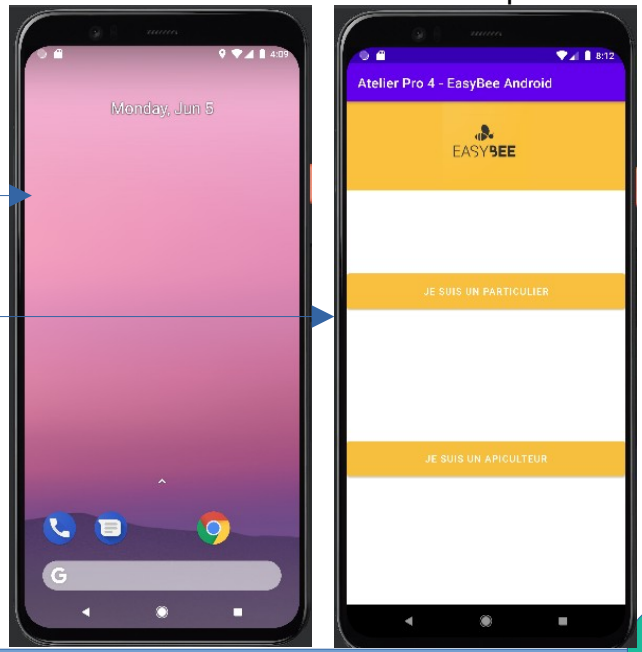


Ce marteau sert à compiler le code.

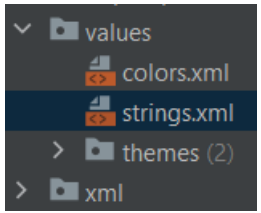
La flèche verte sert à lancer l'application, à lancer seulement lorsque le téléphone est complètement actif.

La flèche blanche sert à lancer le téléphone :

B) Résultat :



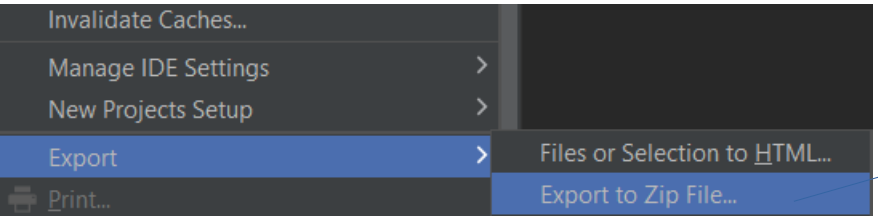
C) Changer le nom de l'app

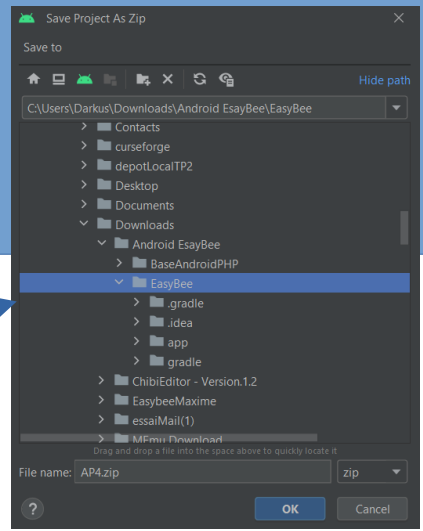



```
<resources>
    <string name="app_name">Atelier Pro 4 - EasyBee Android</string>
</resources>
```

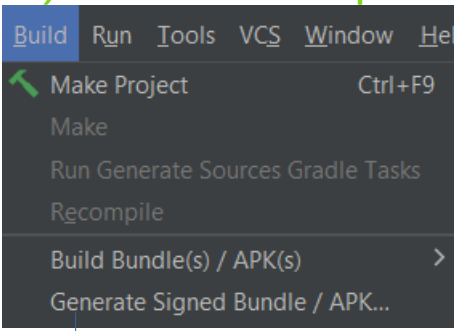
16. Exporter un projet et générer un apk

A) Exporter un projet





B) Générer un apk



Saisir password and confirm sur ses 2 zones et écrire nom et prénom