



Le logiciel
Android studio

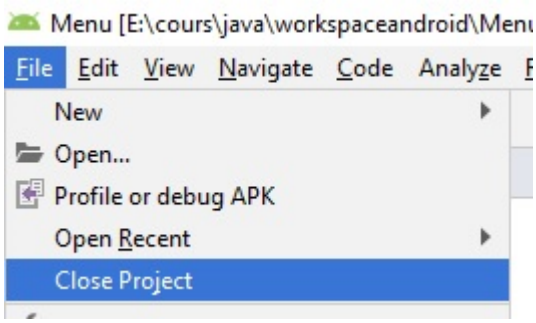




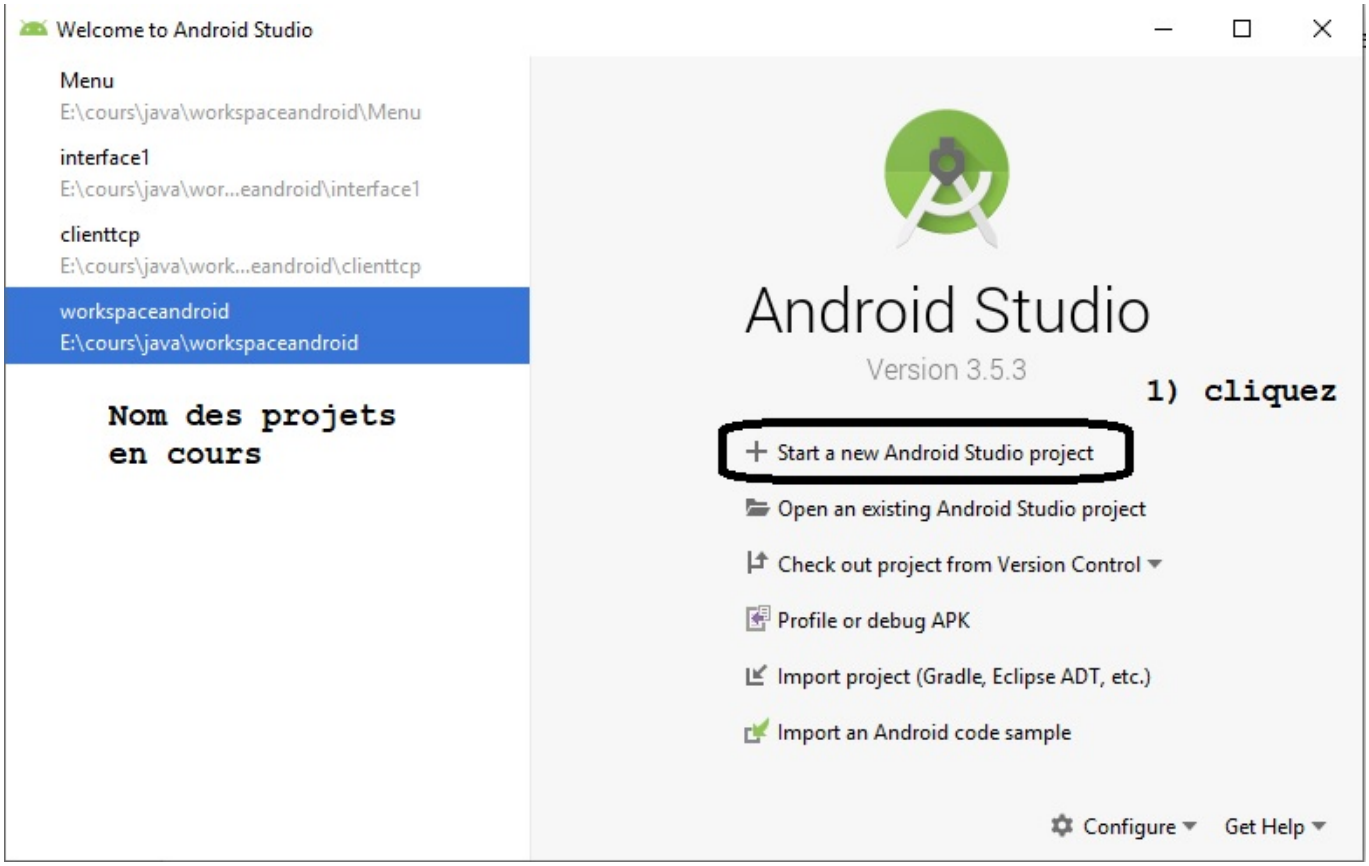
Pour ouvrir un projet double cliquez sur l'icône suivante:



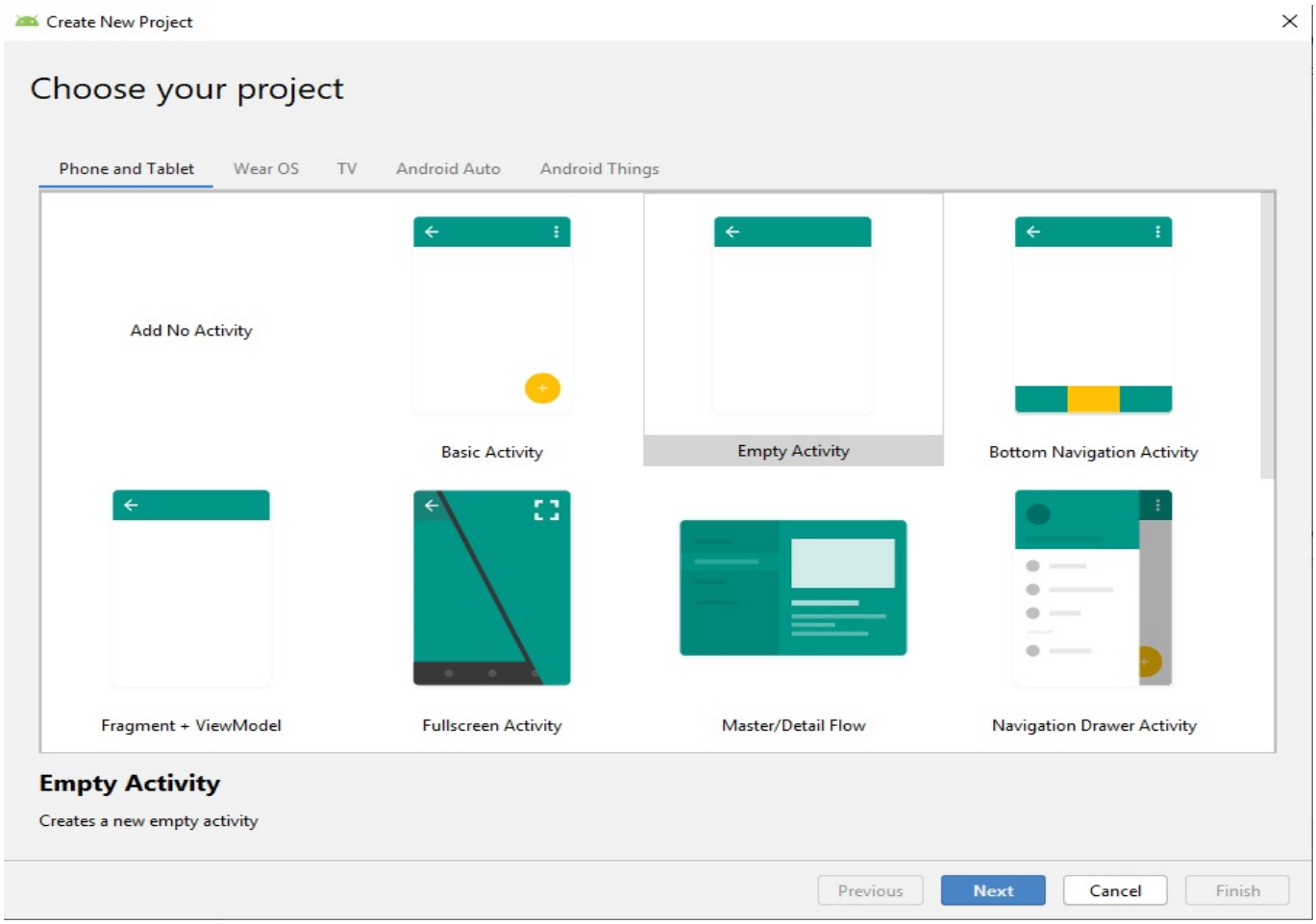
Si vous obtenez directement le lancement du logiciel avec un projet ouvert, vous pouvez fermer ce projet pour en créer un nouveau :



La fenêtre suivante apparait :

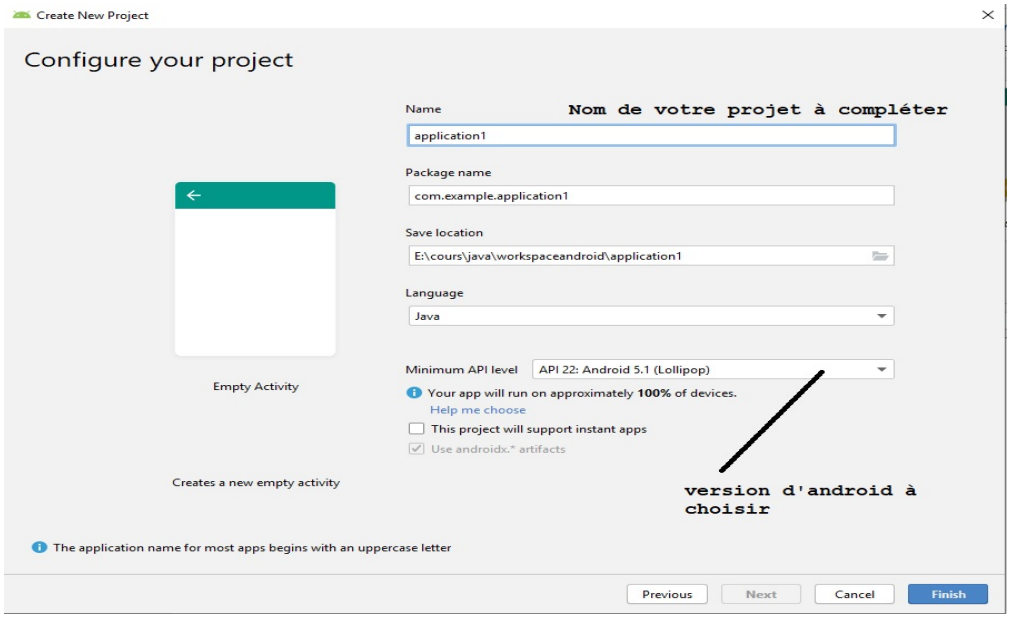


On vous demande sur quel environnement ou plateforme vous souhaitez travailler :



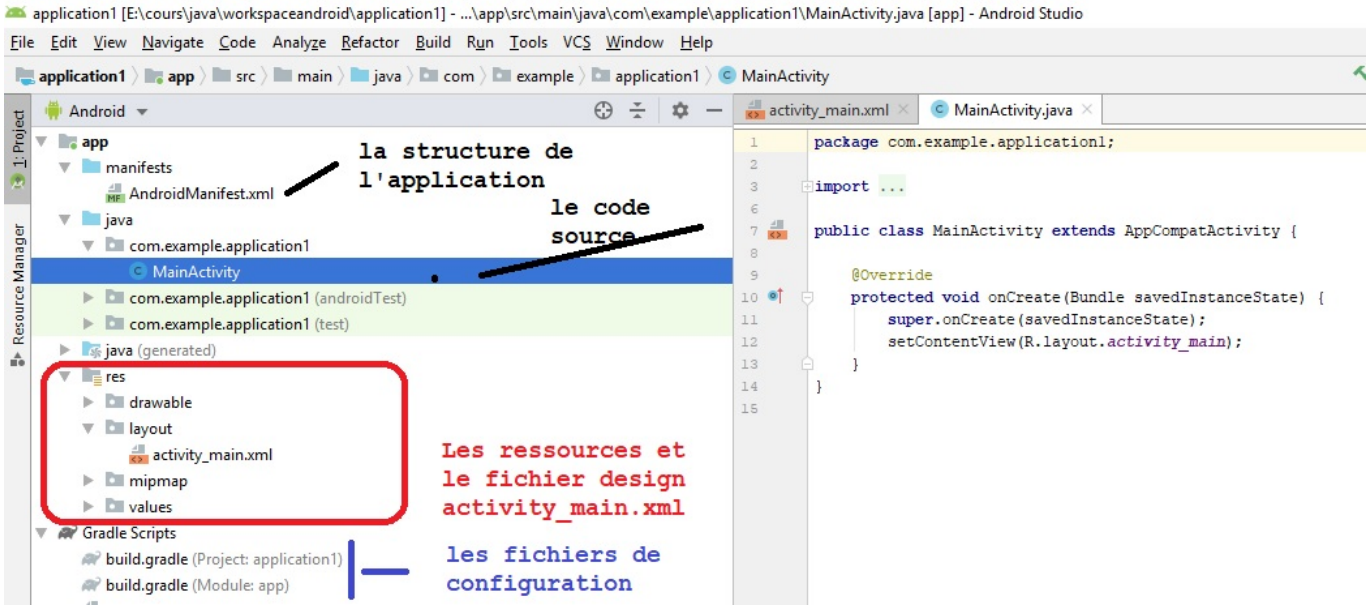
Par défaut l'activité est vide

On vous demande alors le nom du projet ainsi que la version d'Android

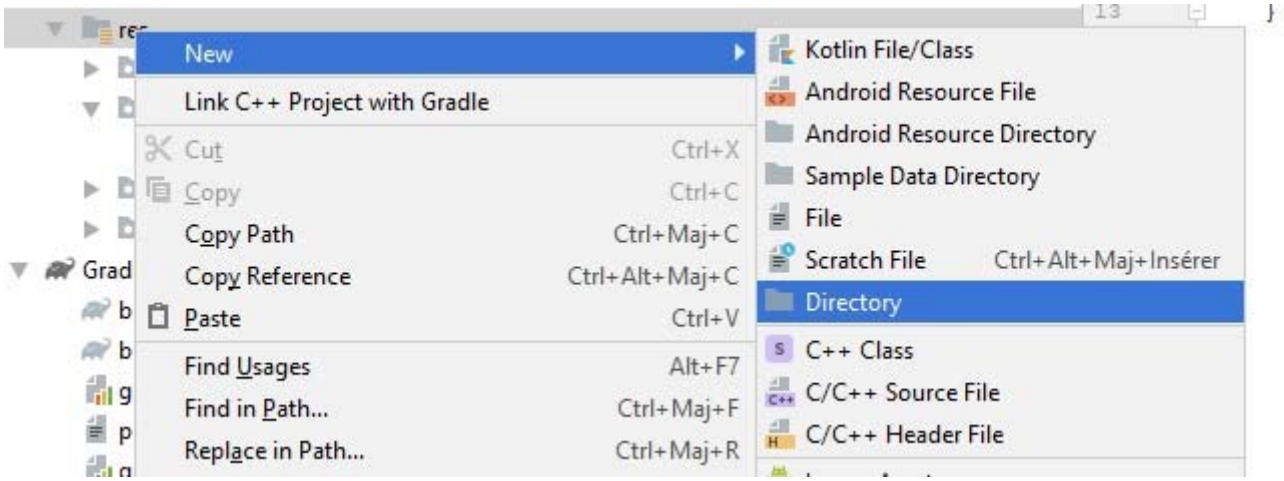


Le téléchargement de la SDK d'android peut se lancer par la suite sur Internet:

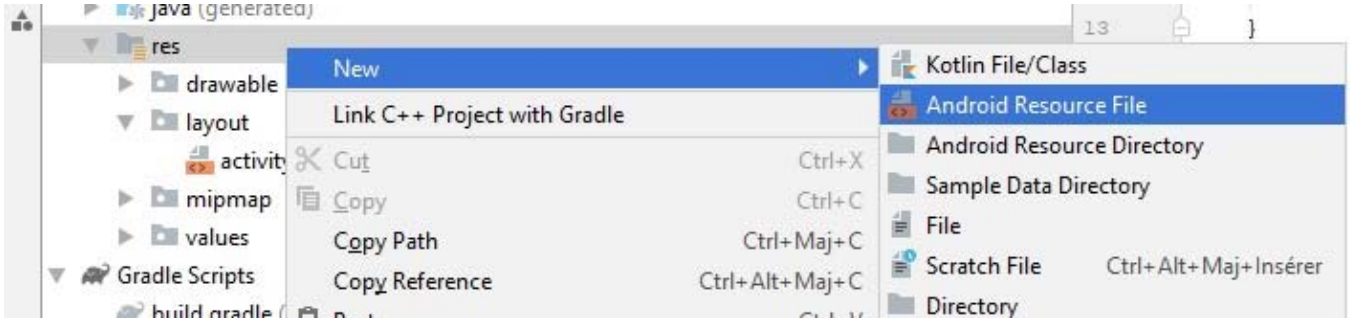
La fenêtre principale apparait :



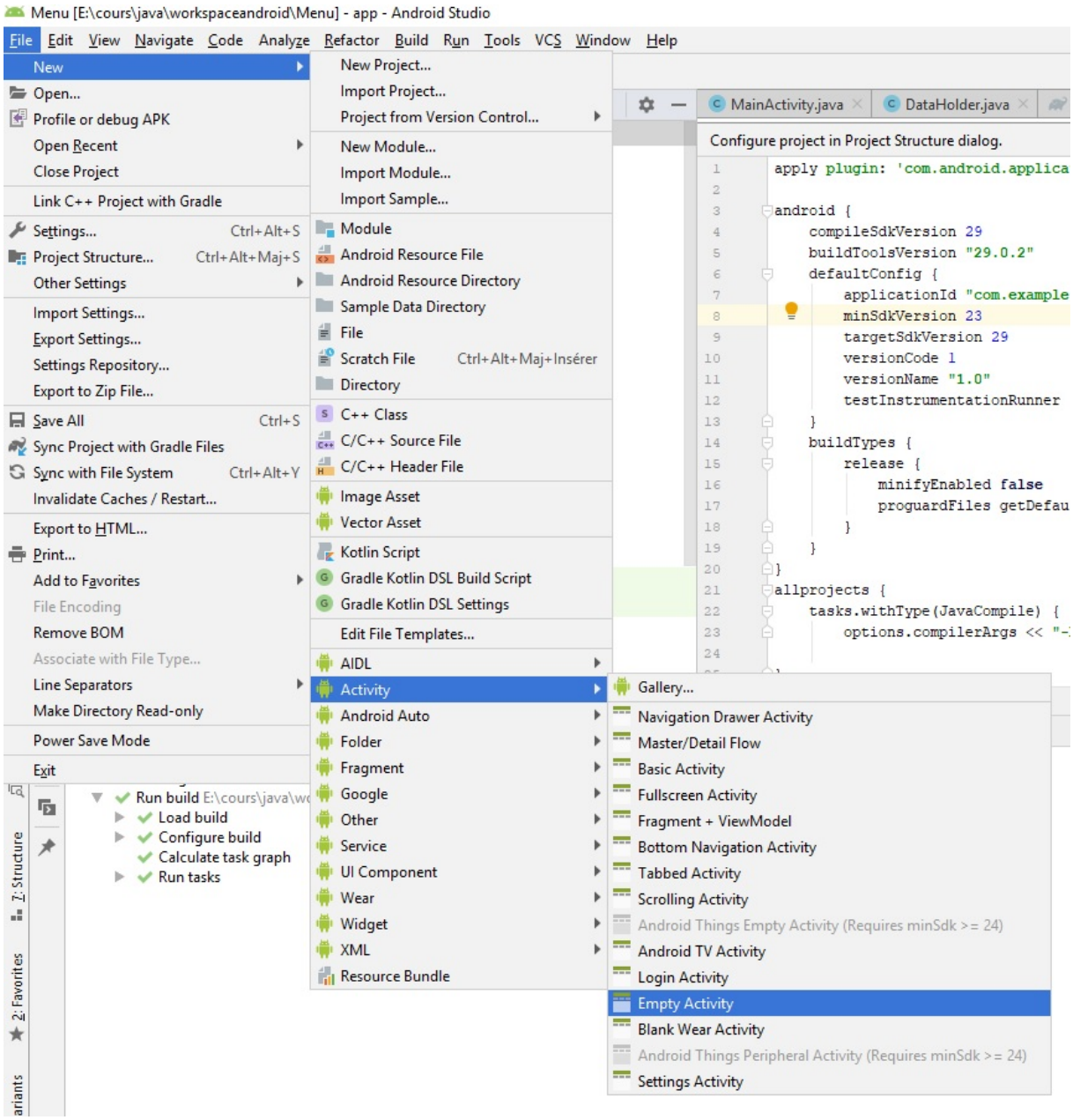
Vous pouvez rajouter un répertoire dans le répertoire principal res pour ajouter une ressource :



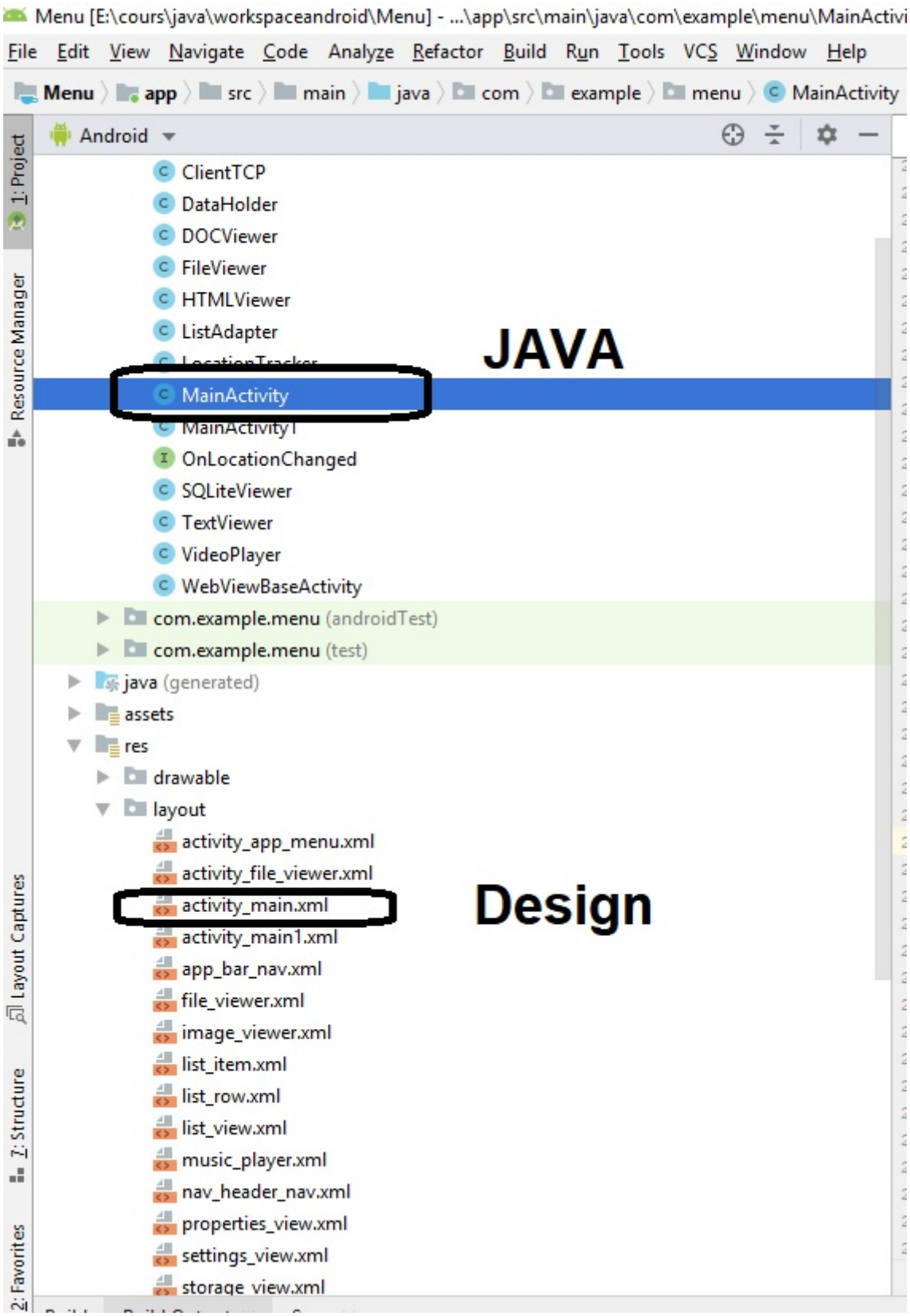
Vous pouvez aussi ajouter un fichier xml de ressource :



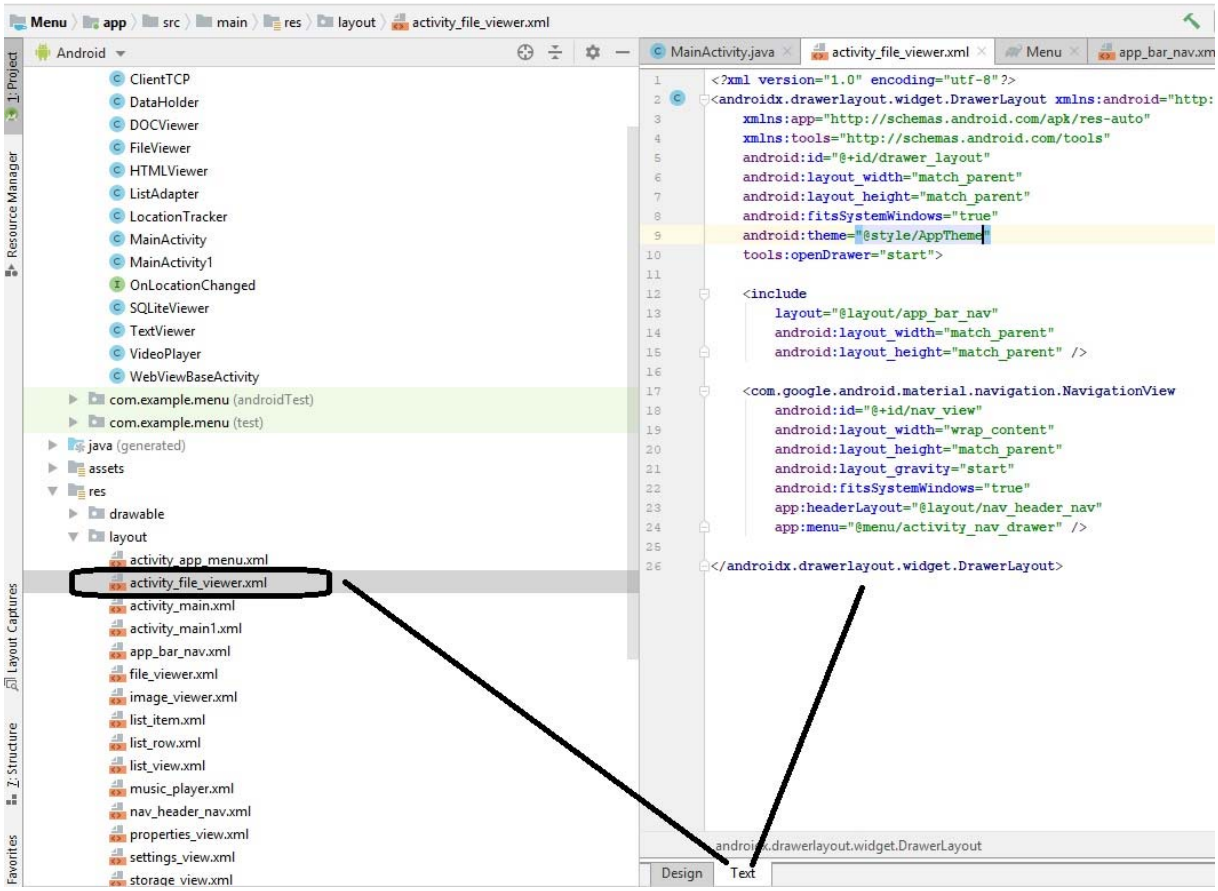
Pour créer une nouvelle activité :



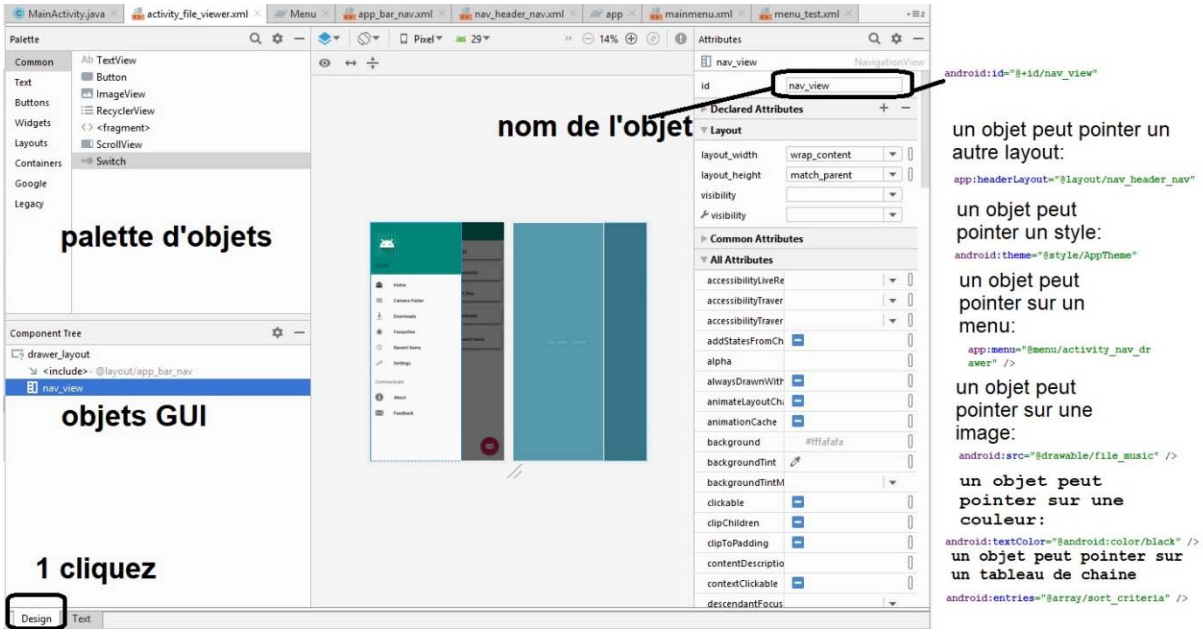
On retrouve le code source java dans APP et dans les ressources le design :



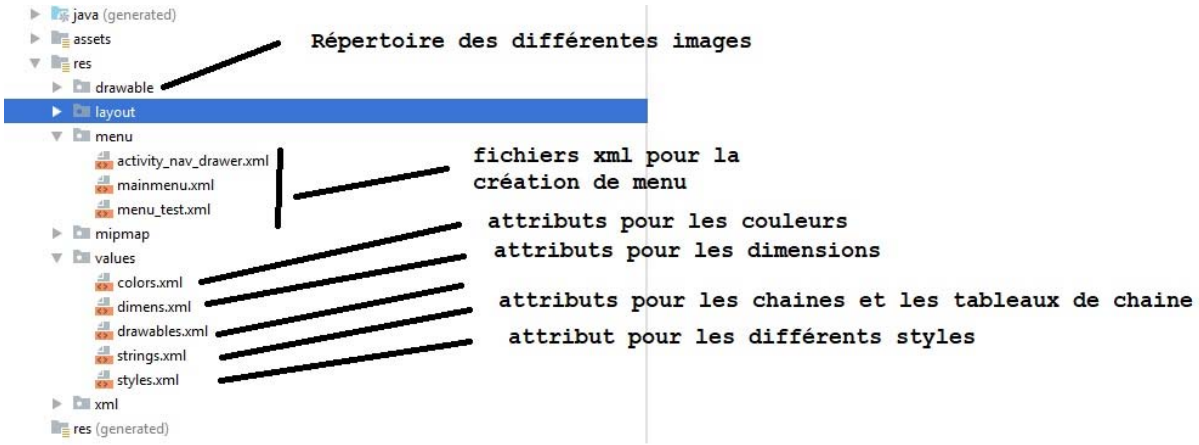
Le fichier xml définit le design de l'IHM dans la rubrique layout des ressources:



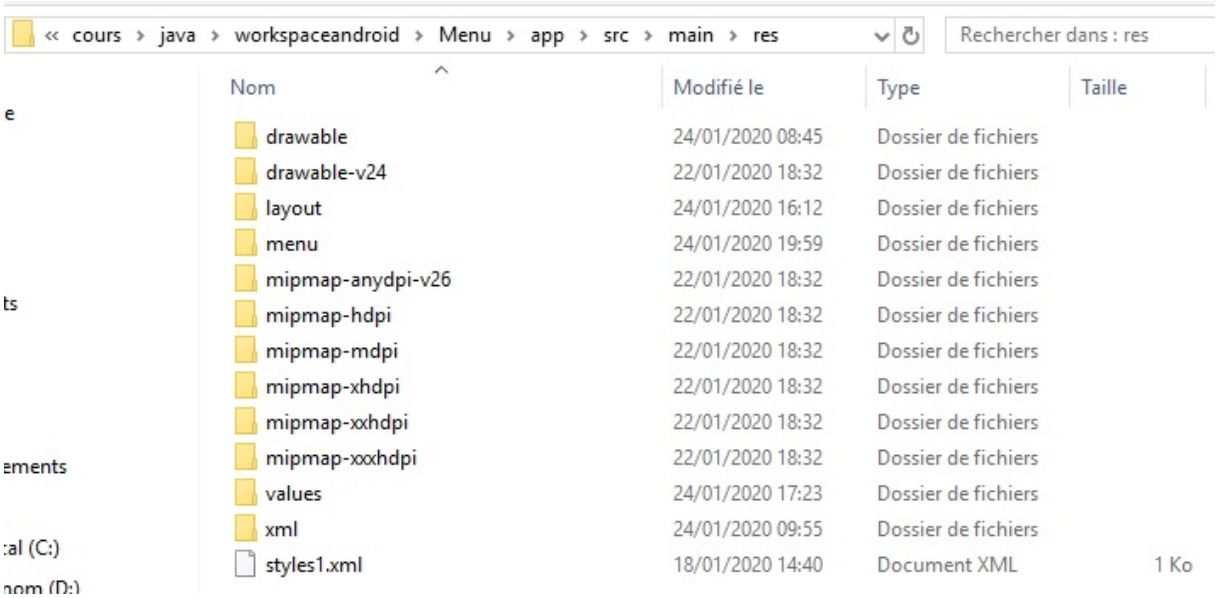
Le design permet de concevoir notre IHM avec les objets GUI et l'inspecteur d'objet à droite :



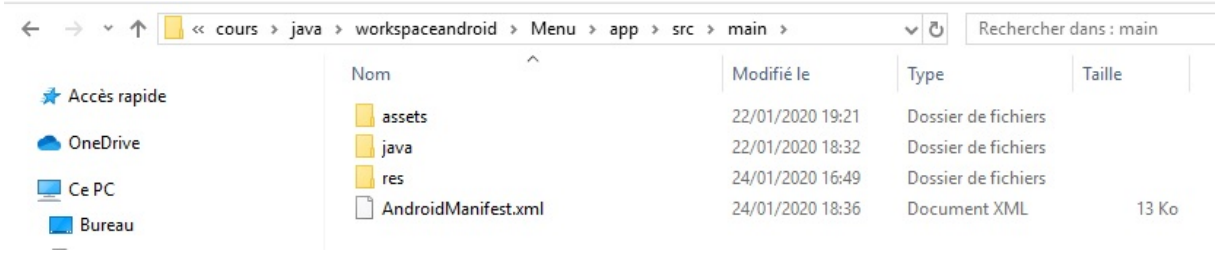
Les différentes ressources sont regroupées suivant une arborescence dans le répertoire res :

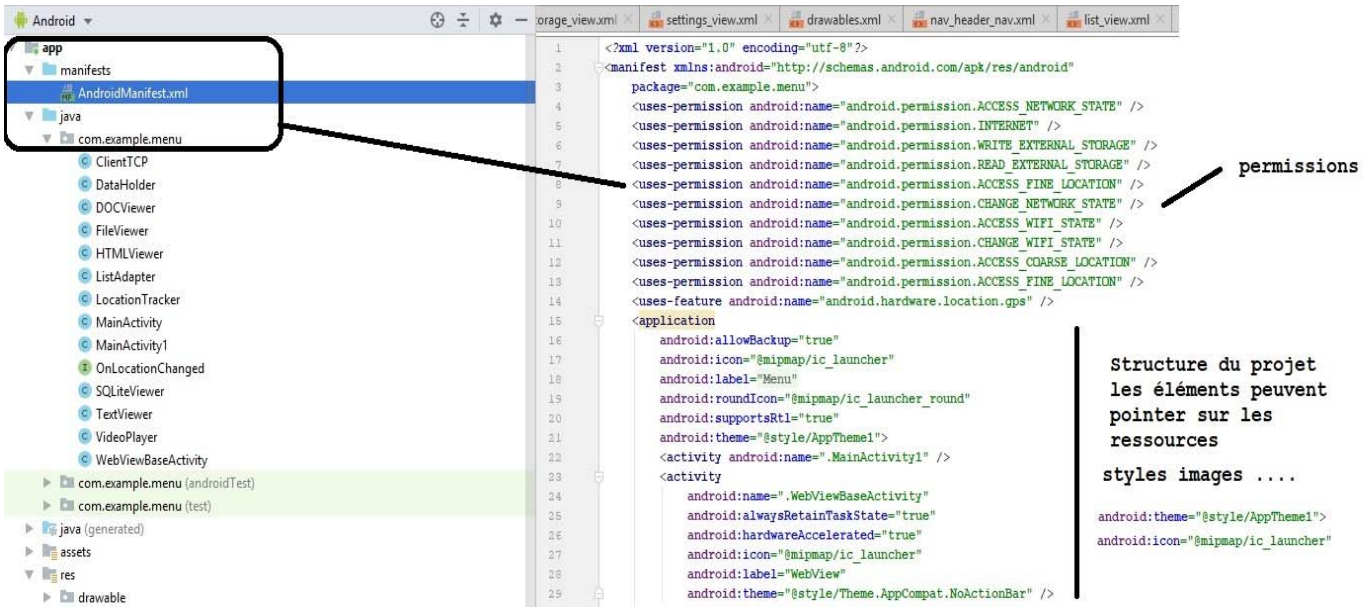


On retrouve ces ressources dans les répertoires suivants :

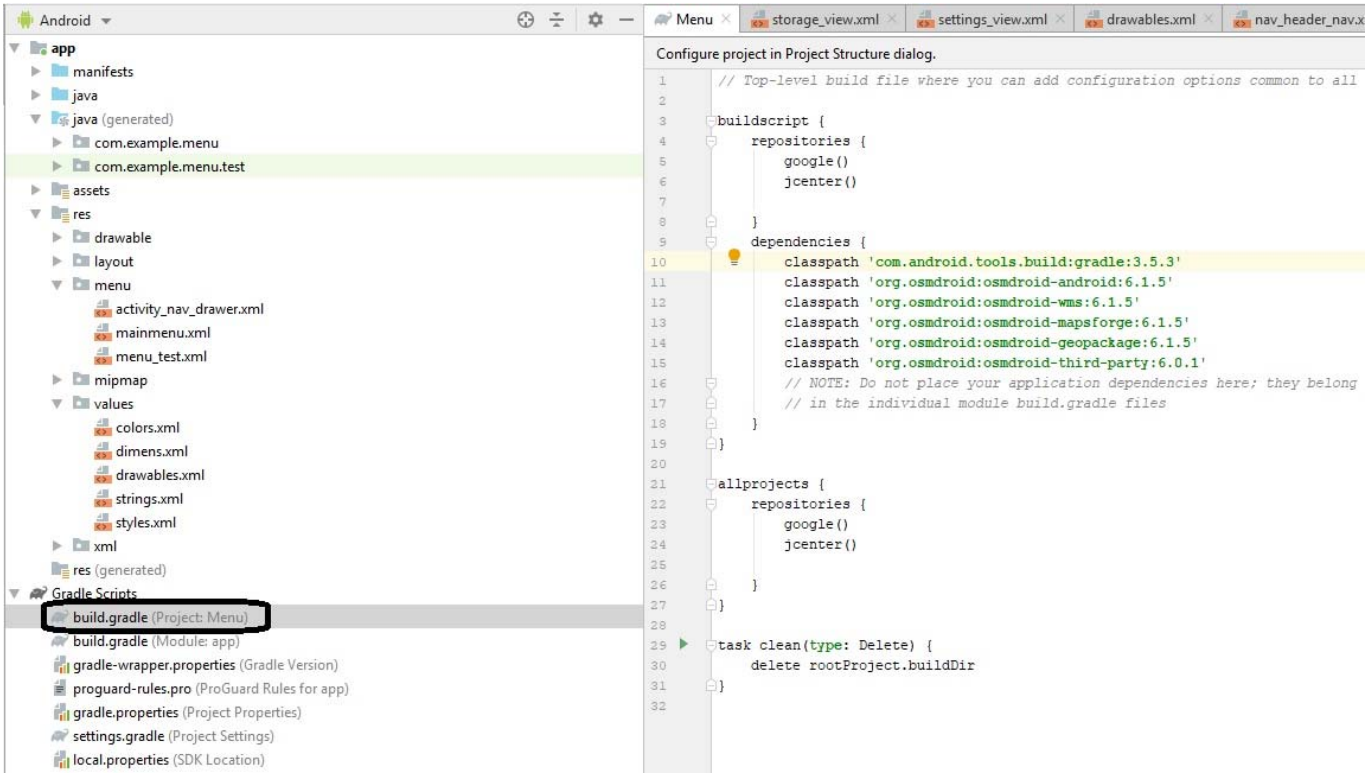


La structure de l'application se situe dans le fichier AndroidManifest.xml :

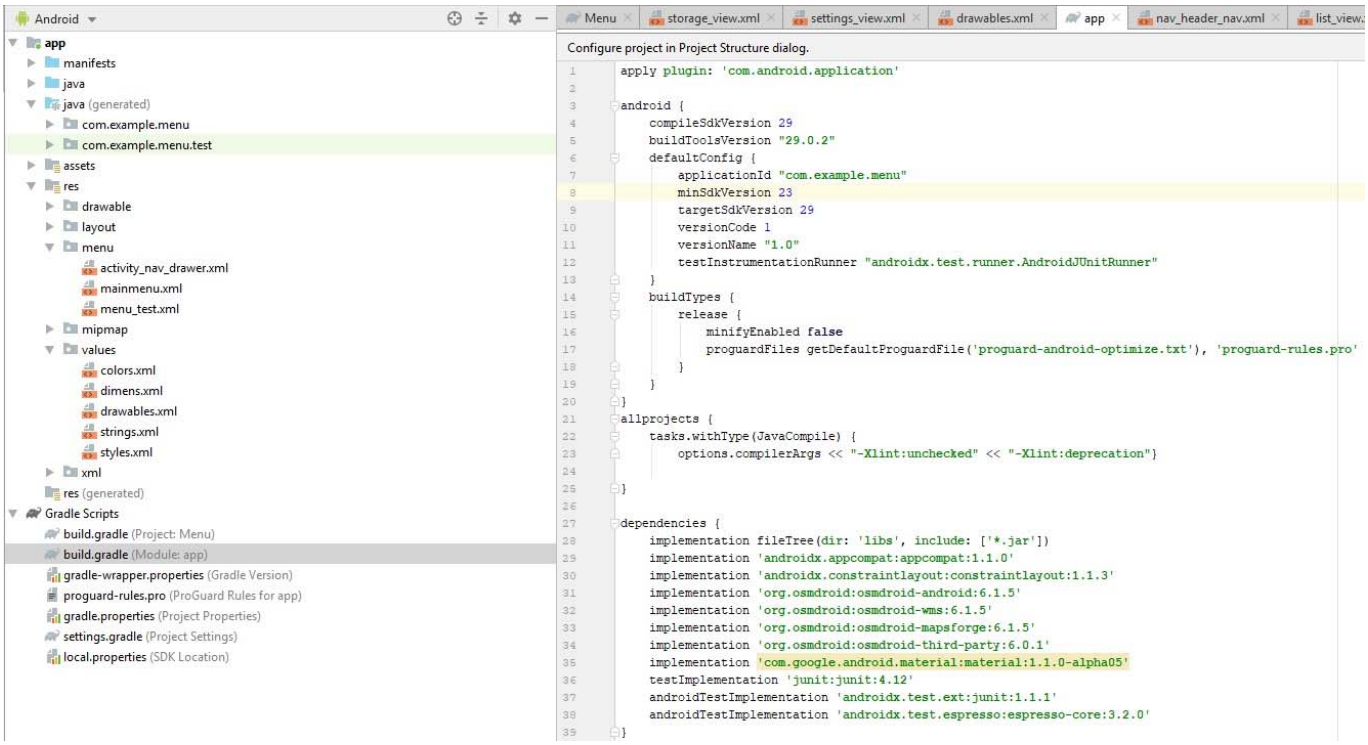


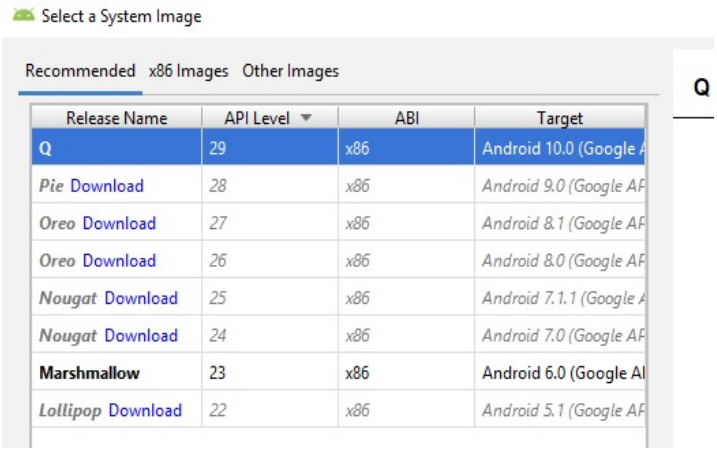


Le fichier build.gradle (java\workspaceandroid\Menu) répertorie toute la dépendance des packages pour la phase de compilation

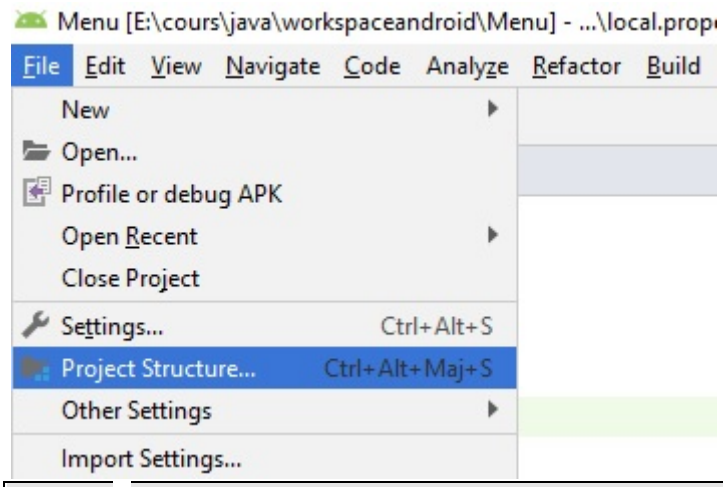


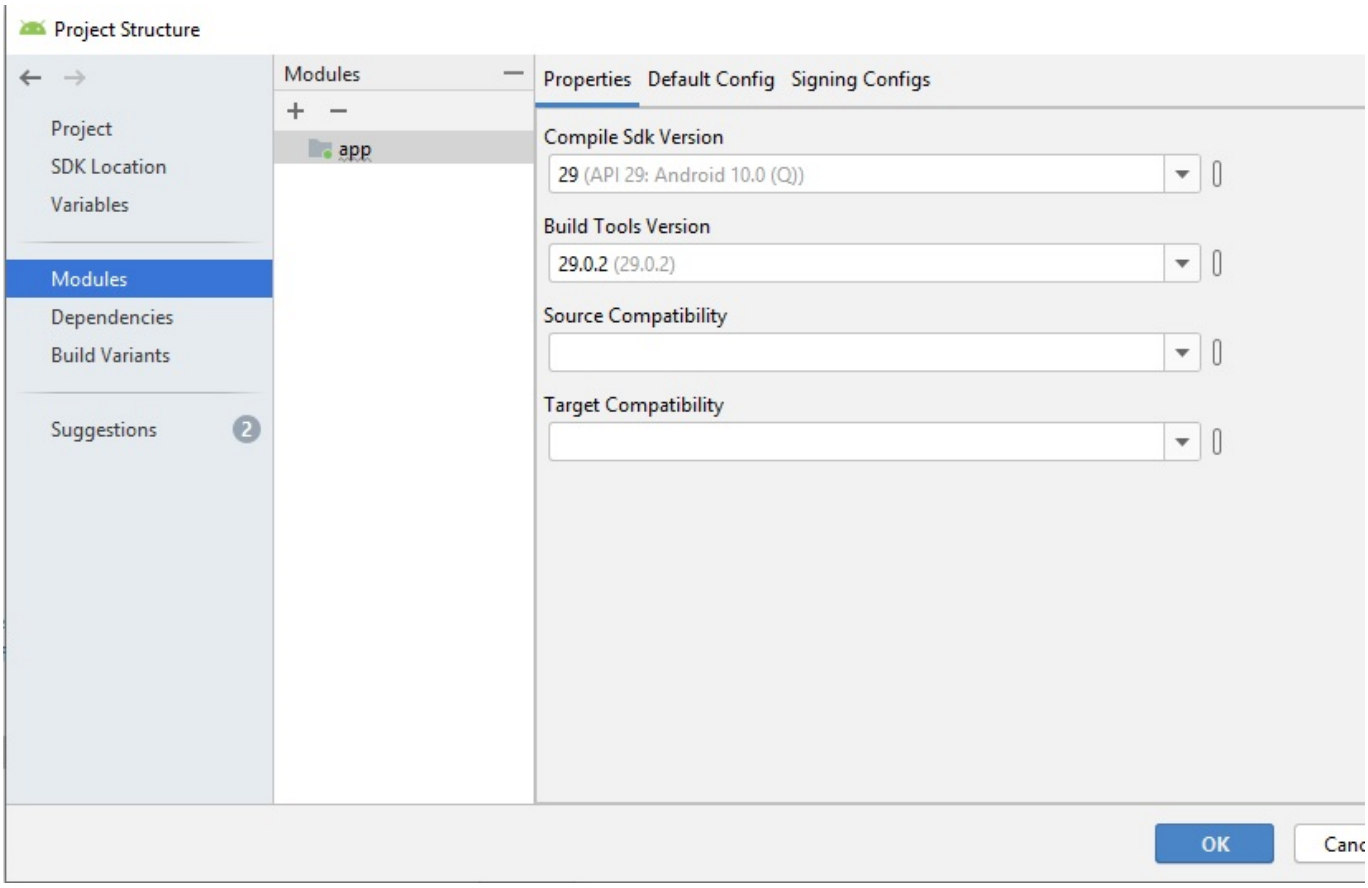
L'autre fichier build.gradle (\java\workspaceandroid\Menu\app) répertorie aussi la dépendance des packages pour la phase de compilation ainsi que le type de sdk choisi pour la compilation et le développement :





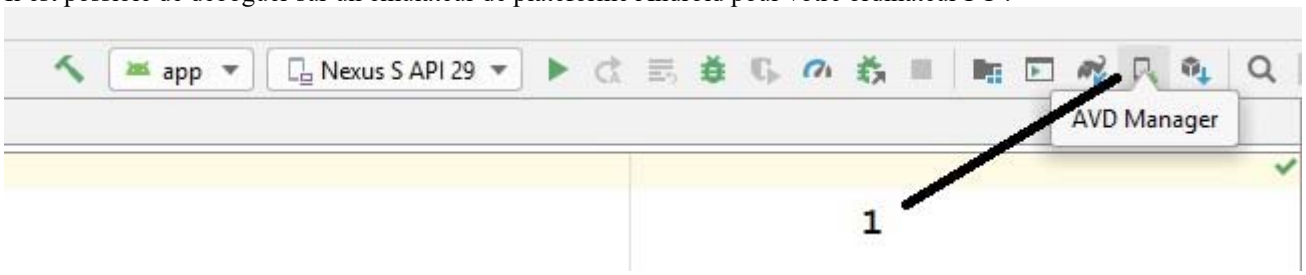
La structure du projet définit dans les fichiers de métadonnées se retrouve dans :



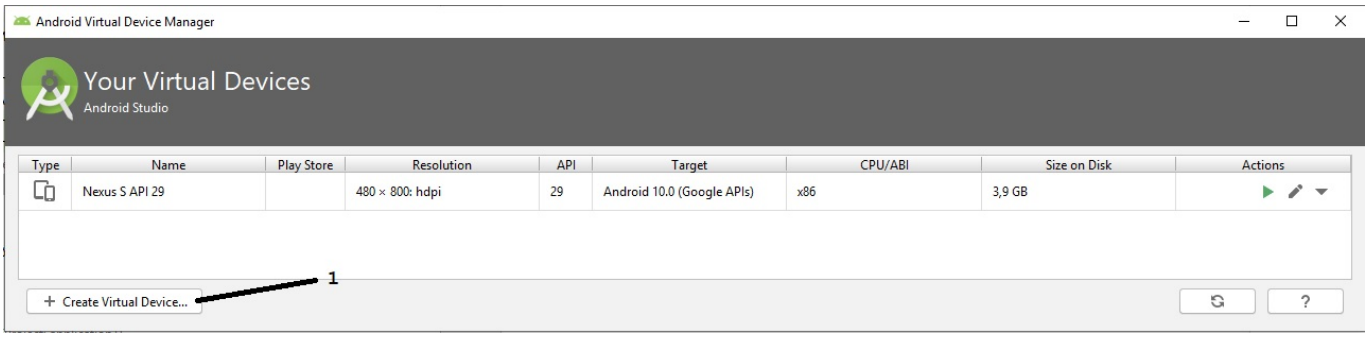


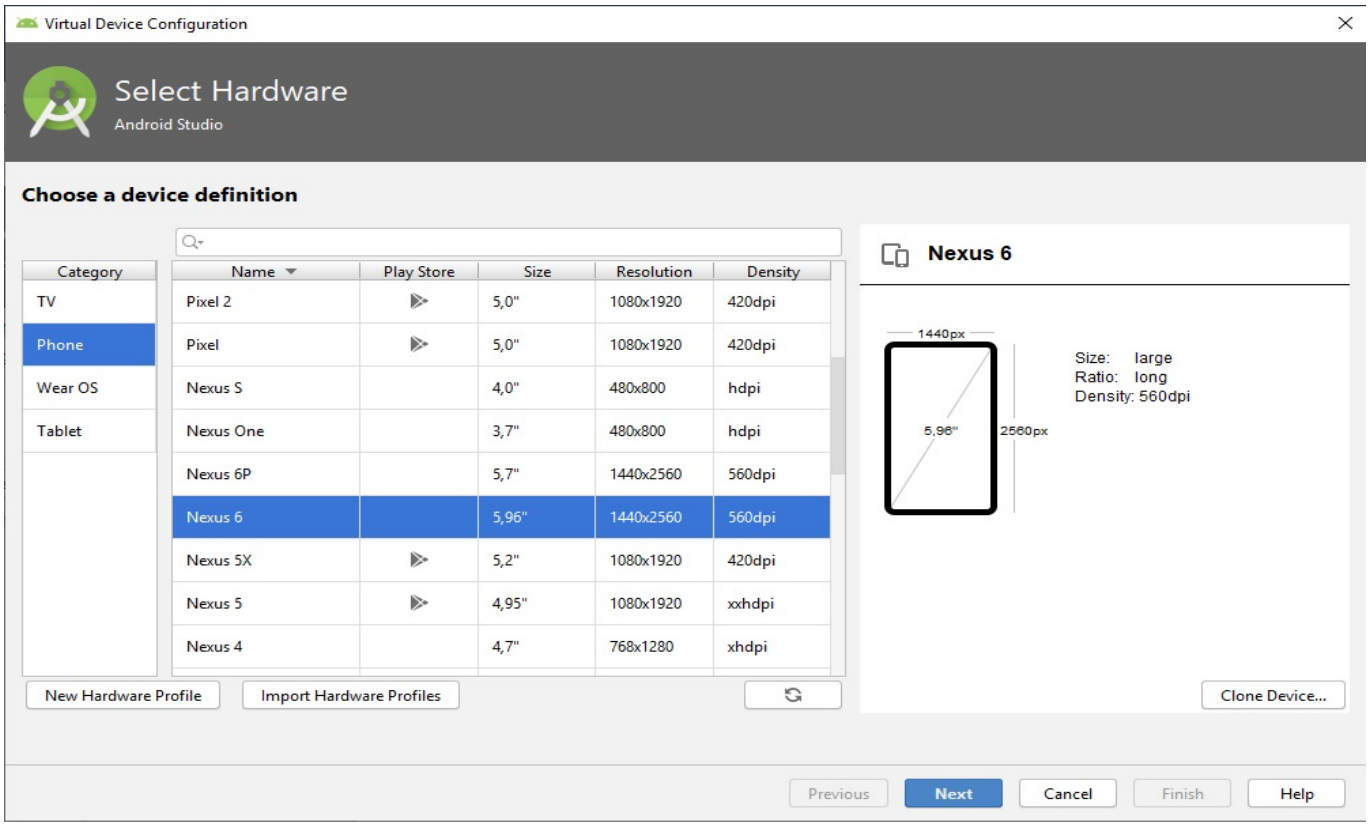
Le débogage

Il est possible de déboguer sur un émulateur de plateforme Android pour votre ordinateur PC :

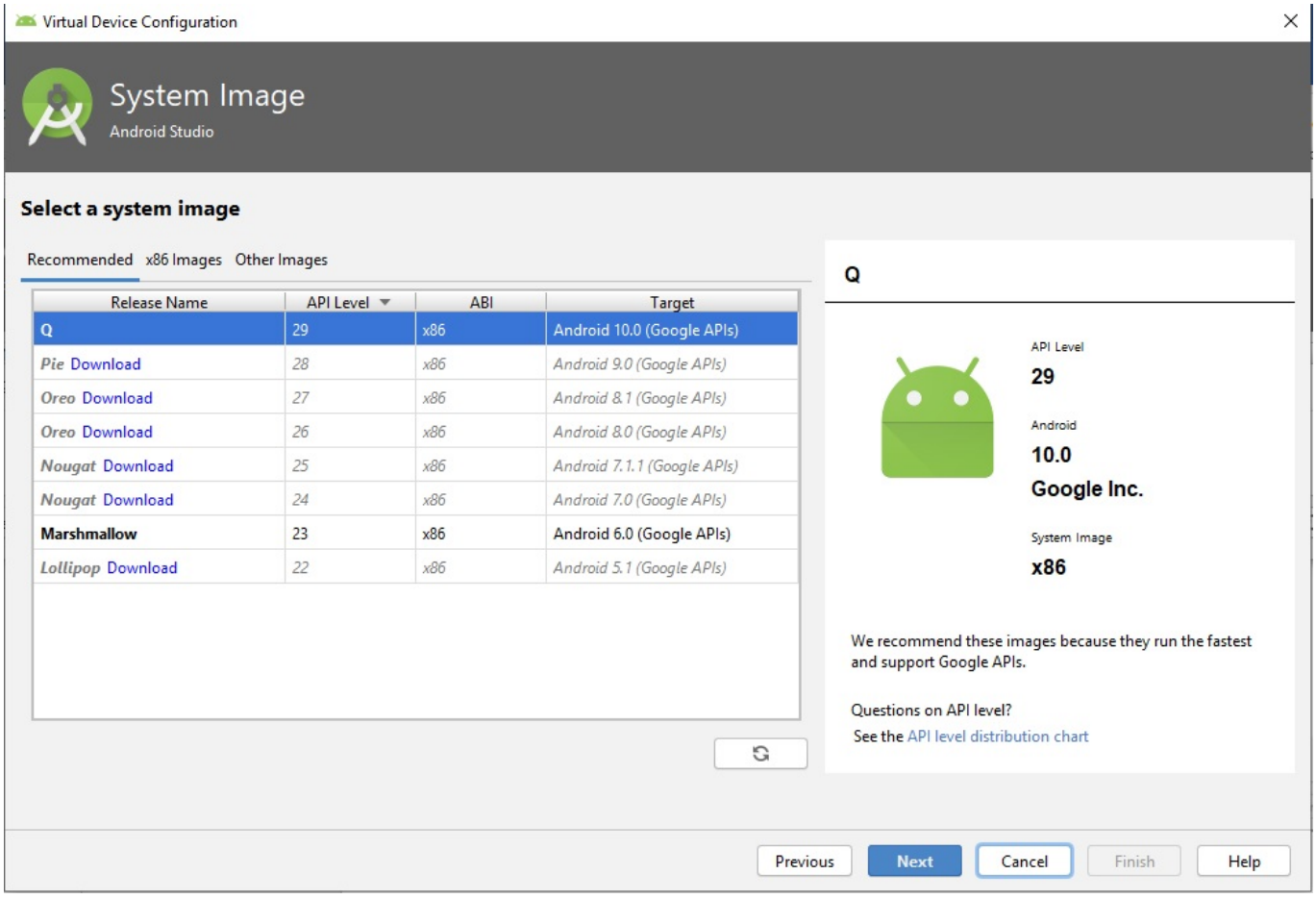


Cliquez sur create Virtual device une configuration est déjà présente :





Puis la version d'android



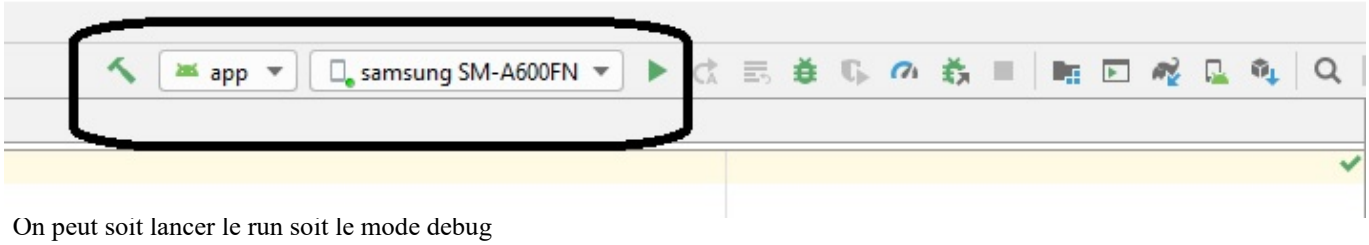
Le debogage sur plateforme virtuel est très gourmand en ressource et plante souvent :

On peut aussi déboguer sur une plateforme existante (Il faudra passer le portable en mode développement).

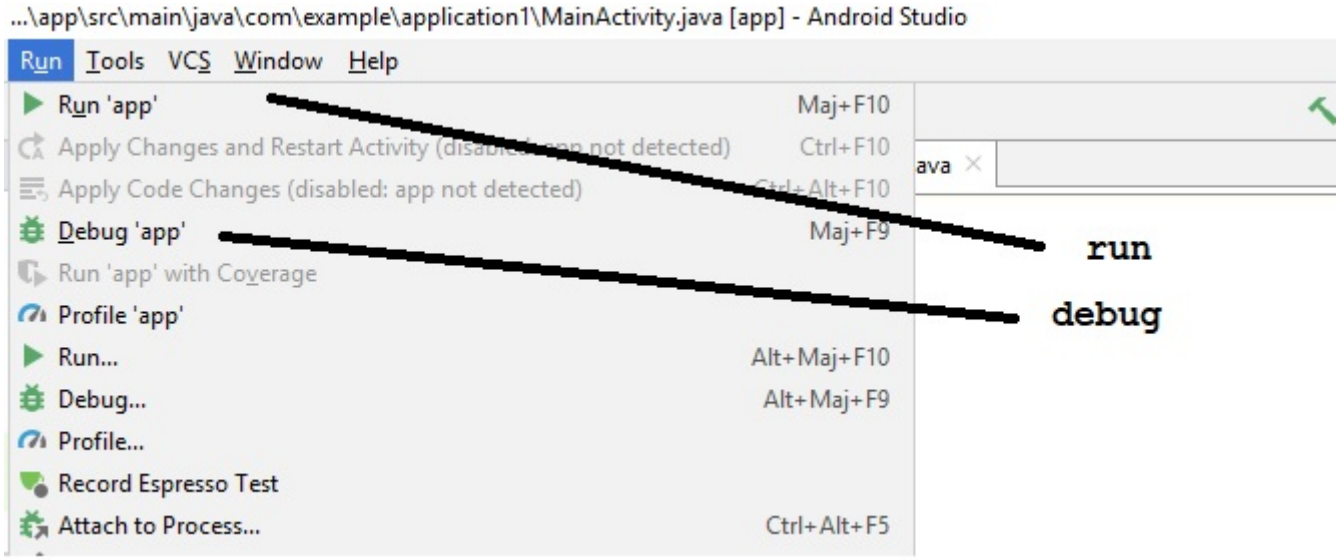
Pour passer en mode développement allez dans paramètre sur votre téléphone puis à propos du telephone → informations sur le logiciel et cliquez 7 fois sur le numéro de version.

Une nouvelle rubrique apparait dans paramètres sur le téléphone option de développement et cochez la case débogage USB. Branchez un câble USB entre votre portable et l'ordinateur.

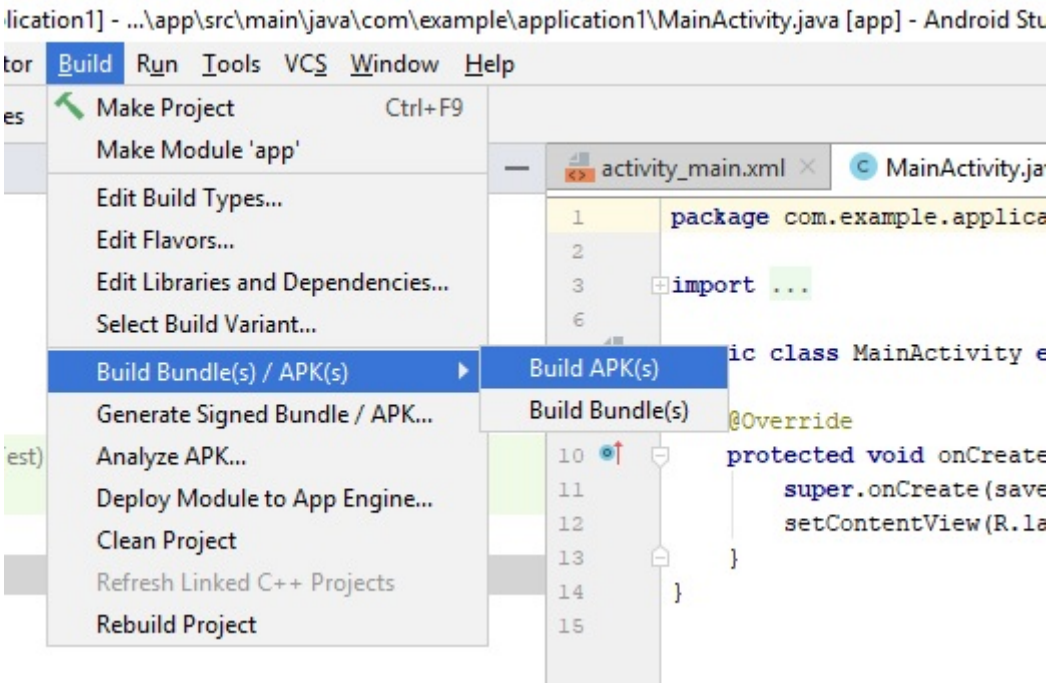
Le menu suivant apparait :



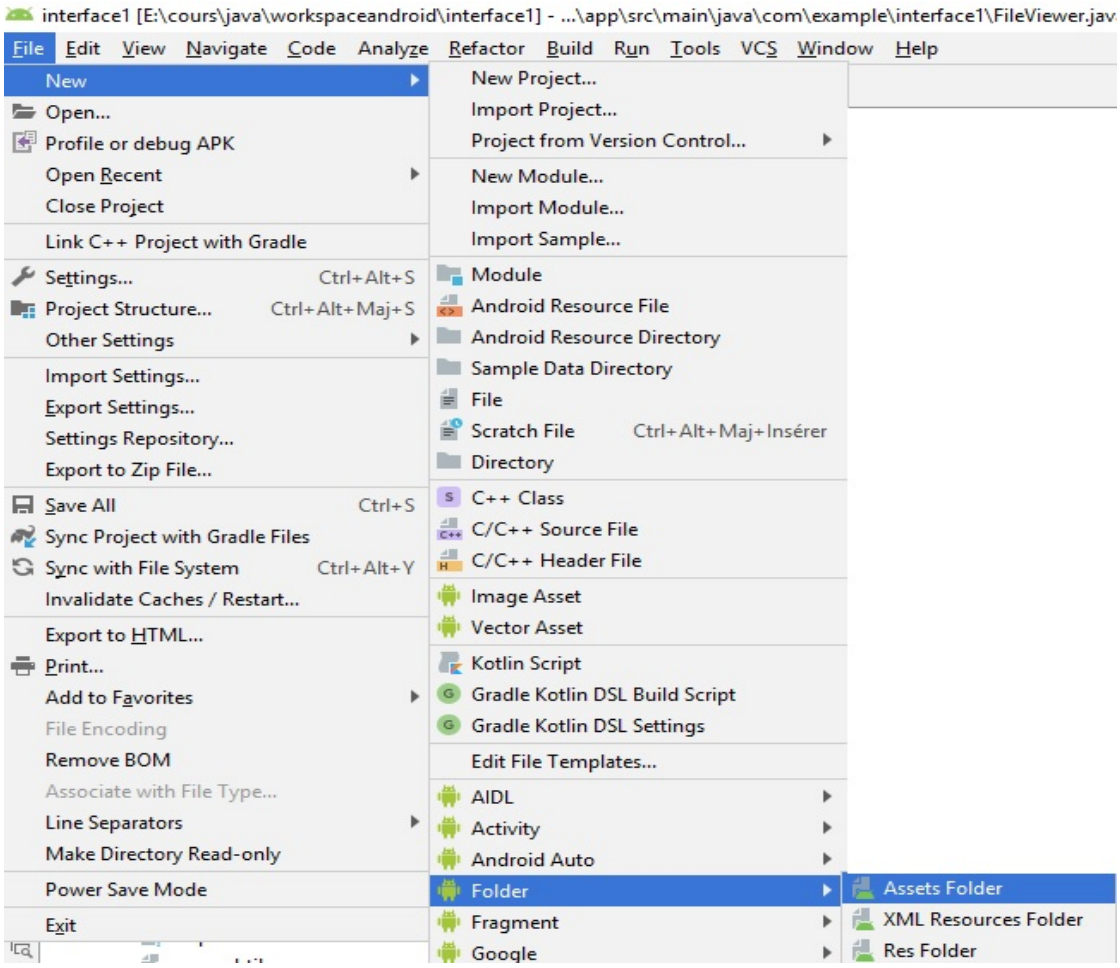
On peut soit lancer le run soit le mode debug

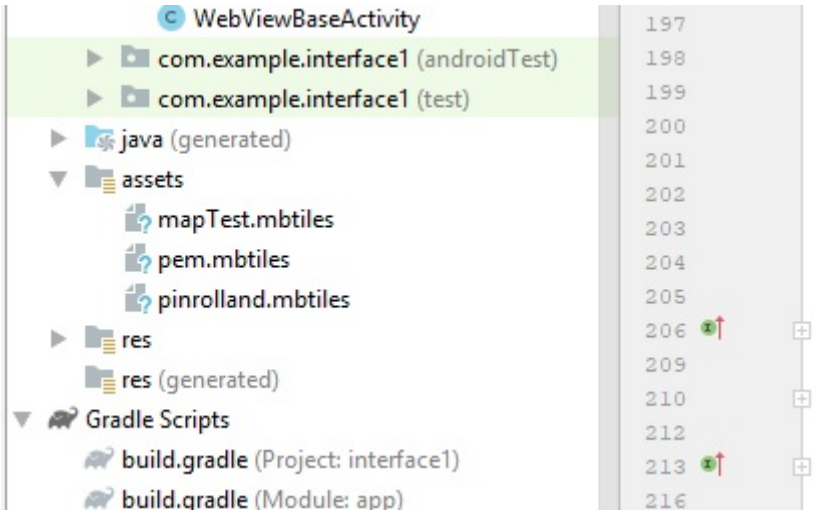


<u>On peut aussi créer un fichier APK pour le déploiement sur notre environnement par la suite :</u>



On peut lors du déploiement ajouter des fichiers multimédias ou autres dans un répertoire assets :





Le code JAVA ira donc lire les fichiers déployés dans le répertoire assets :

```
if (file==null) {


    try {
        file = getFileFromAssets("pin.mbtiles");// nom du fichier à charger
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

Fonctionnement d'une application

Au début, le système Android lance l'activité qui est marquée action=MAIN et catégorie=LAUNCHER dans AndroidManifest.xml.
Ensuite, d'autres activités peuvent être démarrées. Chacune se met « devant » les autres comme sur une pile.
Deux cas sont possibles :
• La précédente activité se termine, on ne revient pas dedans.
Par exemple, une activité où on tape son login et son mot de passe lance l'activité principale et se termine.
• La précédente activité attend la fin de la nouvelle car elle lui demande un résultat en retour.
Exemple : une activité de type liste d'items lance une activité pour éditer un item quand on clique longuement dessus, mais attend la fin de l'édition pour rafraîchir la liste.

Lancement sans attente

Rappel, pour lancer Activ2 à partir de Activ1 :

```
Intent intent = new Intent(this, Activ2.class);
startActivity(intent);
```

On peut demander la terminaison de this après lancement de Activ2 ainsi :

```
Intent intent = new Intent(this, Activ2.class);
startActivity(intent);
finish();
```

finish() fait terminer l'activité courante. L'utilisateur ne pourra pas faire back dessus, car elle disparaît de la pile.

Lancement avec attente de résultat

Le lancement d'une activité avec attente de résultat est plus complexe. Il faut définir un *code d'appel* RequestCode fourni au lancement.

```
private static final int APPEL_ACTIV2 = 1; // RequestCode
Intent intent = new Intent(this, Activ2.class);
startActivityForResult(intent, APPEL_ACTIV2);
```

Ce code identifie l'activité lancée, afin de savoir plus tard que c'est d'elle qu'on revient. Par exemple,
On pourrait lancer au choix plusieurs activités : édition, copie, suppression d'informations. Il faut pouvoir les distinguer au retour.

Lancement avec attente, suite

Ensuite, il faut définir une méthode *callback* qui est appelée lorsqu'on revient dans notre activité :

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
// uti a fait back
if (resultCode == Activity.RESULT_CANCELED) return;
// selon le code d'appel
switch (requestCode) {
case APPEL_ACTIV2: // on revient de Activ2
...
}
}
```

Méthode onActivityResult

Quand on revient dans l'activité appelante, Android lui fait exécuter cette méthode :
onActivityResult(int requestCode, int resultCode, Intent data)
• requestCode est le code d'appel de startActivityForResult
• resultCode vaut soit RESULT_CANCELED soit RESULT_OK, voir le transparent précédent
• data est fourni par l'activité appelée et qui vient de se terminer.
Ces deux dernières viennent d'un appel à setResult(resultCode, data)

```
@Override
public void onBackPressed(){// appelée lors de la fermeture de l'activité par le bouton retour
    super.onBackPressed();/*éffectue une animation de transition si y a en, avant de finir l'Activity*/
   /* int a=1;
    intentsaisie.putExtra("typeAff", a);
    setResult(RESULT_OK, intentsaisie);*/
}
```

Terminaison d'une activité

L'activité lancée par la première peut se terminer pour deux raisons :
• Volontairement, en appelant la méthode finish() :
setResult(RESULT_OK);
finish();
• À cause du bouton « back » du téléphone, son action revient à faire ceci :
setResult(RESULT_CANCELED);
finish();
Dans ces deux cas, on revient dans l'activité appelante (sauf si elle-même avait fait finish().

Transport d'informations dans un Intent

Les Intent servent aussi à transporter des informations d'une activité à l'autre : les *extras*.
Voici comment placer des données dans un Intent :

```
Intent intent =
new Intent(this, DeleteInfoActivity.class);
intent.putExtra("idInfo", idInfo);
intent.putExtra("hiddencopy", hiddencopy);
startActivity(intent);
```

putExtra(nom, valeur) rajoute un couple (nom, valeur) dans l'intent. La valeur doit être *sérialisable* : nombres, chaînes et structures simples.

Extraction d'informations d'un Intent

Ces instructions récupèrent les données d'un Intent :

```
Intent intent = getIntent();
Integer idInfo = intent.getIntExtra("idInfo", -1);
bool hidden = intent.getBooleanExtra("hiddencopy", false);
```

• getIntent() retourne l'Intent qui a démarré cette activité.
• get*Type*Extra(*nom*, *valeur par défaut*) retourne la valeur de ce nom si elle en fait partie, la valeur par défaut sinon.

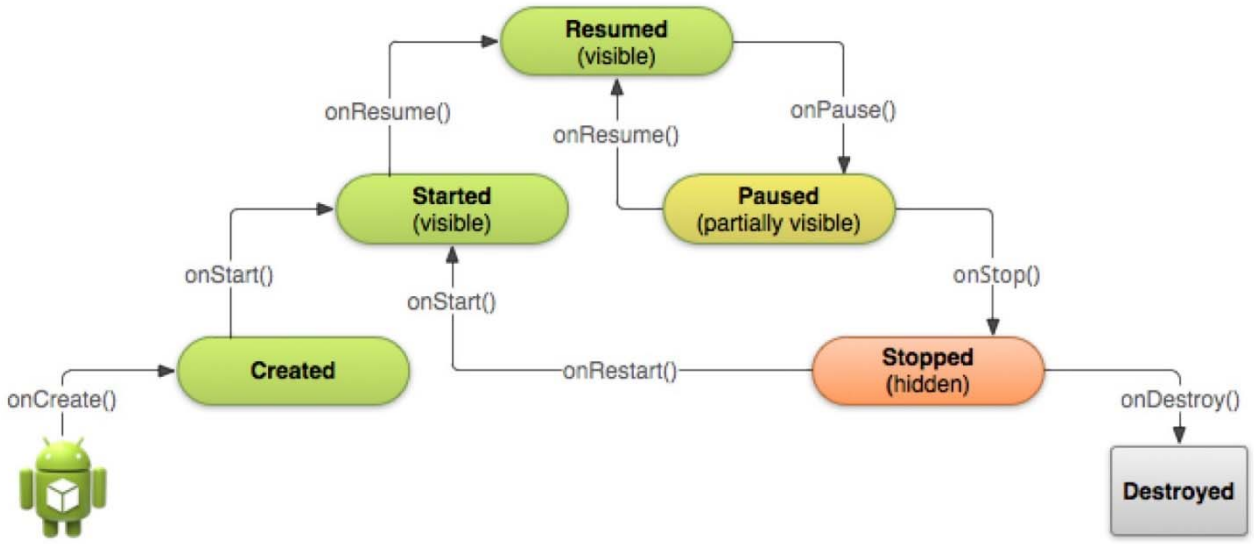
Il est très recommandé de placer les chaînes dans des constantes, dans la classe appelée :

```
public static final String EXTRA_IDINFO = "idInfo";
public static final String EXTRA_HIDDEN = "hiddencopy";
```

Cycle de vie d'une activité

Voyons maintenant comment fonctionnent les activités.
• Démarrage (à cause d'un Intent)
• Apparition/masquage sur écran
• Terminaison

Une activité se trouve dans l'un de ces états :
• active (*resumed*) : elle est sur le devant, l'utilisateur peut jouer avec,
• en pause (*paused*) : partiellement cachée et inactive, car une autre activité est venue devant,
• stoppée (*stopped*) : totalement invisible et inactive, ses variables sont préservées mais elle ne tourne plus.



Événements de changement d'état

La classe Activity reçoit des événements de la part du système Android, ça appelle des fonctions appelées *callbacks*.

Exemples :

onCreate Un Intent arrive dans l'application, il déclenche la création d'une activité, dont l'interface.

onPause Le système prévient l'activité qu'une autre activité est passée devant, il faut enregistrer les informations au cas où l'utilisateur ne revienne pas.

Terminaison d'une activité

Voici la prise en compte de la terminaison définitive d'une activité, avec la fermeture d'une base de données :

```
@Override
public void onDestroy() {
// obligatoire
super.onDestroy();
// fermer la base
db.close();
}
```

Pause d'une activité

Cela arrive quand une nouvelle activité passe devant, exemple : un appel téléphonique. Il faut libérer les ressources qui consomment de l'énergie (animations, GPS. . .).

```
@Override

public void onPause() {// méthode appelée lors de la mise en pause de l'activité
super.onPause();
// arrêter les animations sur l'écran
...
}
@Override

 public void onResume() {// méthode appelée lors du retour vers l'activité
super.onResume();
// démarrer les animations
...
}
```

Remarque pour la cartographie et l'accès aux fichiers

Pour les sdk supérieur ou égal à 29 rajoutez en rouge dans manifest.xml:

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme1"
    android:requestLegacyExternalStorage="true"
    >
```

Une configuration type pour des listview

Pour le thème Dans android_manifest.xls

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:value="splitActionBarWhenNarrow"
    android:theme="@style/AppTheme">
```

Dans le fichier styles.xml relatif au thème

```
<style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">

    <!-- Customize your theme here. -->
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent</item>
    <item name="windowActionBar">true</item>
    <item name="windowNoTitle">false</item>
    <!-- <item name="windowActionBar">true</item>-->

</style>
```

Dans le fichier activity_main.xml

```
<ListView
        android:id="@android:id/list"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```

Dans le fichier list.xml

```
<?xml version="1.0" encoding="utf-8"?>
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@android:id/text1"

    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceListItemSmall"
    android:gravity="center_vertical"
    android:paddingStart="?android:attr/listPreferredItemPaddingStart"
    android:paddingEnd="?android:attr/listPreferredItemPaddingEnd"
    android:minHeight="?android:attr/listPreferredItemHeightSmall" />
```

Dans le répertoire menu menu_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item android:id="@+id/menu_refresh"
        android:title="menu_refresh"
        android:checkable="false"
        android:orderInCategory="1"
        app:showAsAction="never"/>
    <item android:id="@+id/menu_environnement"
        android:title="menu_environnement"
        app:showAsAction="never"/>

     </menu>
```

Si `app:showAsAction="never"/>` alors le sous menu est dessous les 3 petits points
Si app:showAsAction="ifRoom|withText"/> alors le menus se trouve dans la barre des menus

Dans la main_activity.java pour la liste et la barre de menus

```
public class MainActivity extends AppCompatActivity
```

dans la méthode onCreate

```
pour le titre
setTitle("bluetoothLE");
mAdapter = new BeaconAdapter(this);
 listView = (ListView) findViewById(android.R.id.list);
listView.setAdapter(mAdapter);
// mListView.setAdapter(adapter);
mAdapter.setNotifyOnChange(true);


public void onItemClick( AdapterView<?> parent, View view, int position, long id ) {
        TemperatureBeacon serveur = (TemperatureBeacon) getListView().getItemAtPosition( position
);
        mAdapter.remove(serveur);
        mAdapter.notifyDataSetChanged();
                    }
} );
```

pour le menu :

```
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu_main, menu);
```

pour les événements :

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
       /* case R.id.menu_celsius:
            mAdapter.setCelsius(true);
            break;*/
        case R.id.menu_environnement:
```

```
            Intent intent3 = new Intent(MainActivity.this, EnvironnementActivity.class);
            startActivity(intent3);
            return true;

    }

    return true;
}

protected void onResume() {
    super.onResume();
    // Initializes list view adapter.
        //setListAdapter(mAdapter);
    scanLeDevice(true);
    startScan();
}
```

dans le build.gradle :

```
dependencies {
   // classpath 'com.android.tools.build:gradle:3.5.3'
   // classpath 'com.android.tools.build:gradle:2.2.3'
    classpath 'com.android.tools.build:gradle:3.6.2'
    // NOTE: Do not place your application dependencies here; they belong
    // in the individual module build.gradle files
}
```

Dans le graddle-wrapper.properties

```
distributionUrl=https\://services.gradle.org/distributions/gradle-5.6.4-all.zip
```

Pour effacer un élément de la liste