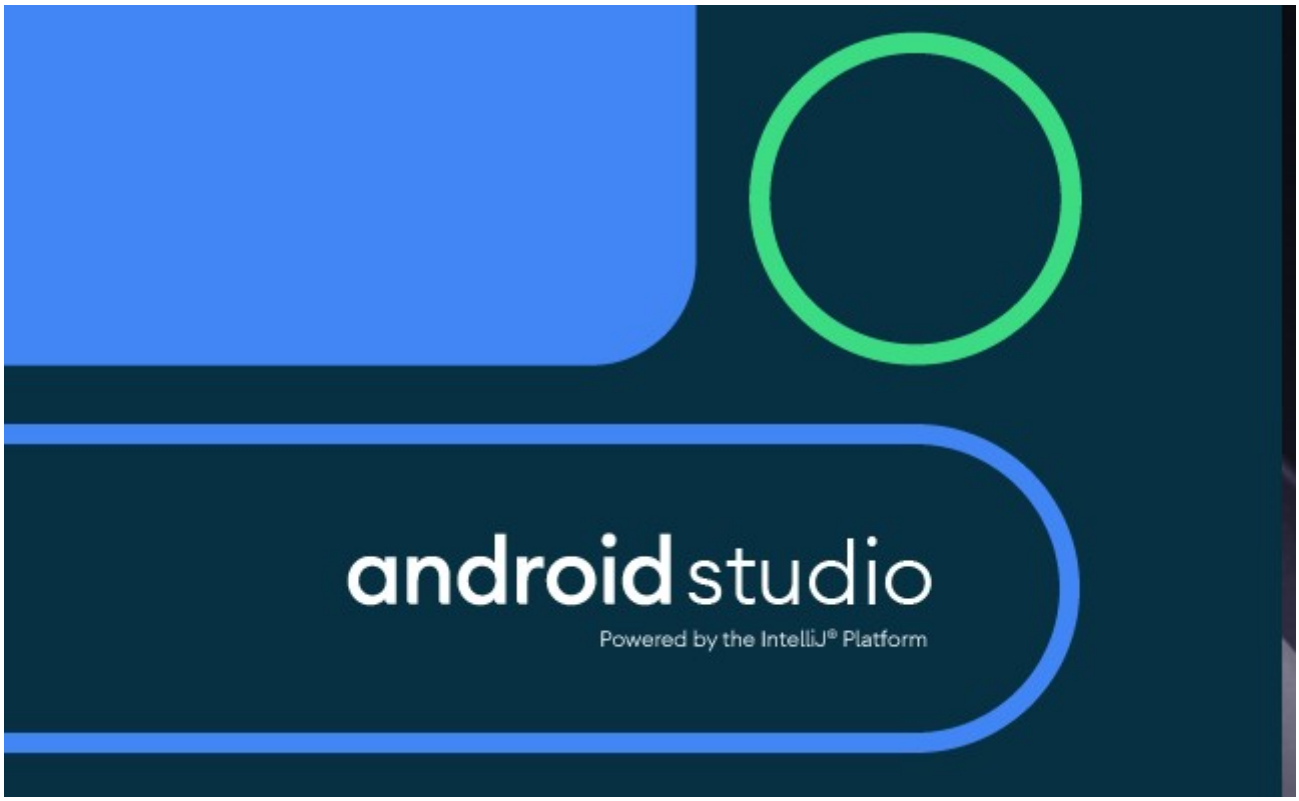
1 Création d'un projet Androïde Studio

Un projet Androïde contient tous les fichiers qui composent le code source de votre application Androïde. Pour créer un nouveau projet Androïde Studio (AS) procéder comme suit :

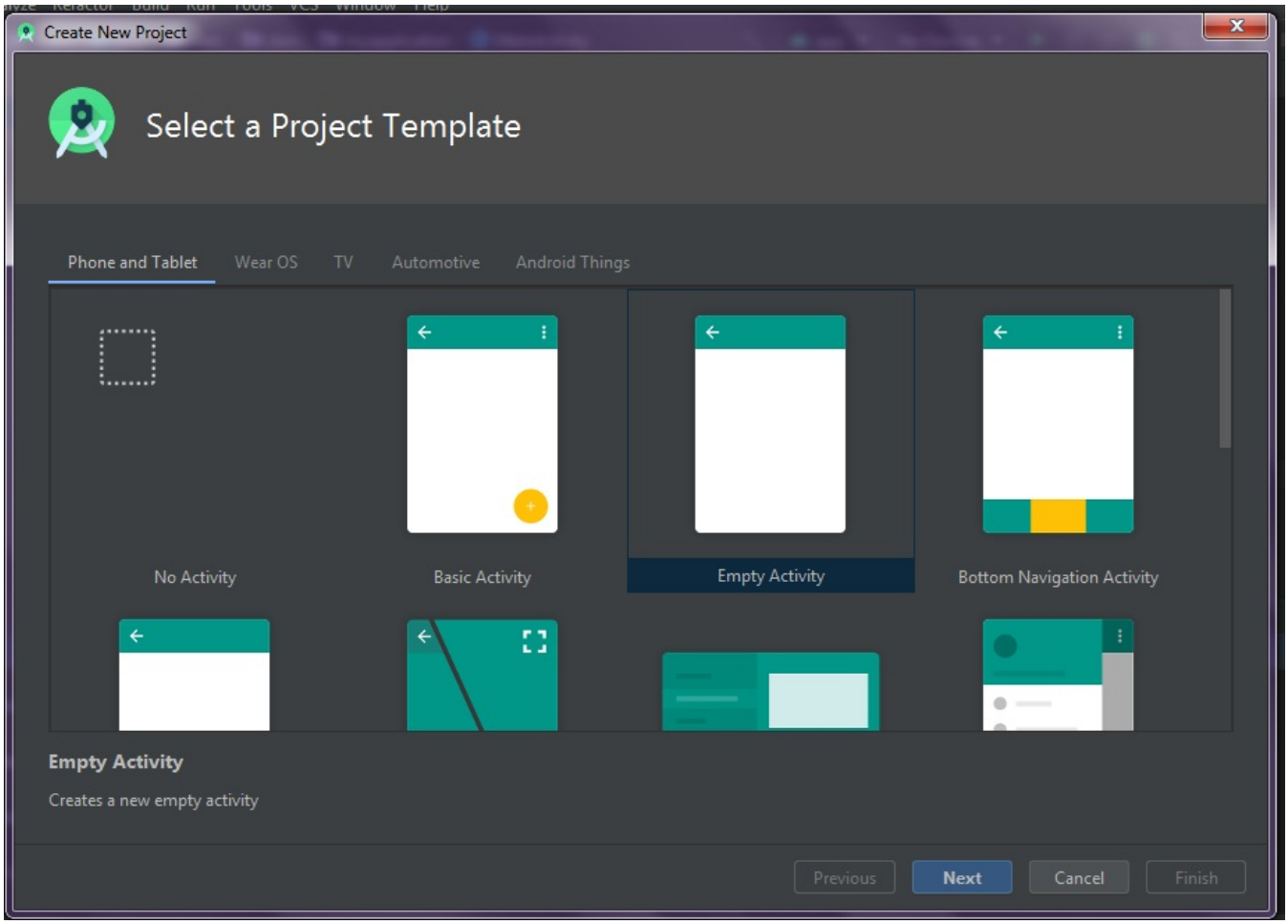
Dans Androïde Studio, créez un nouveau projet :

- Si vous avez un projet déjà ouvert, sélectionnez Nouveau projet à partir du fichier menu.
- Lancer Androïde Studio (AS), la fenêtre « *Welcome*» s'affiche. Cette fenêtre vous permet de créer, d'importer ou d'ouvrir de nouveaux projets.

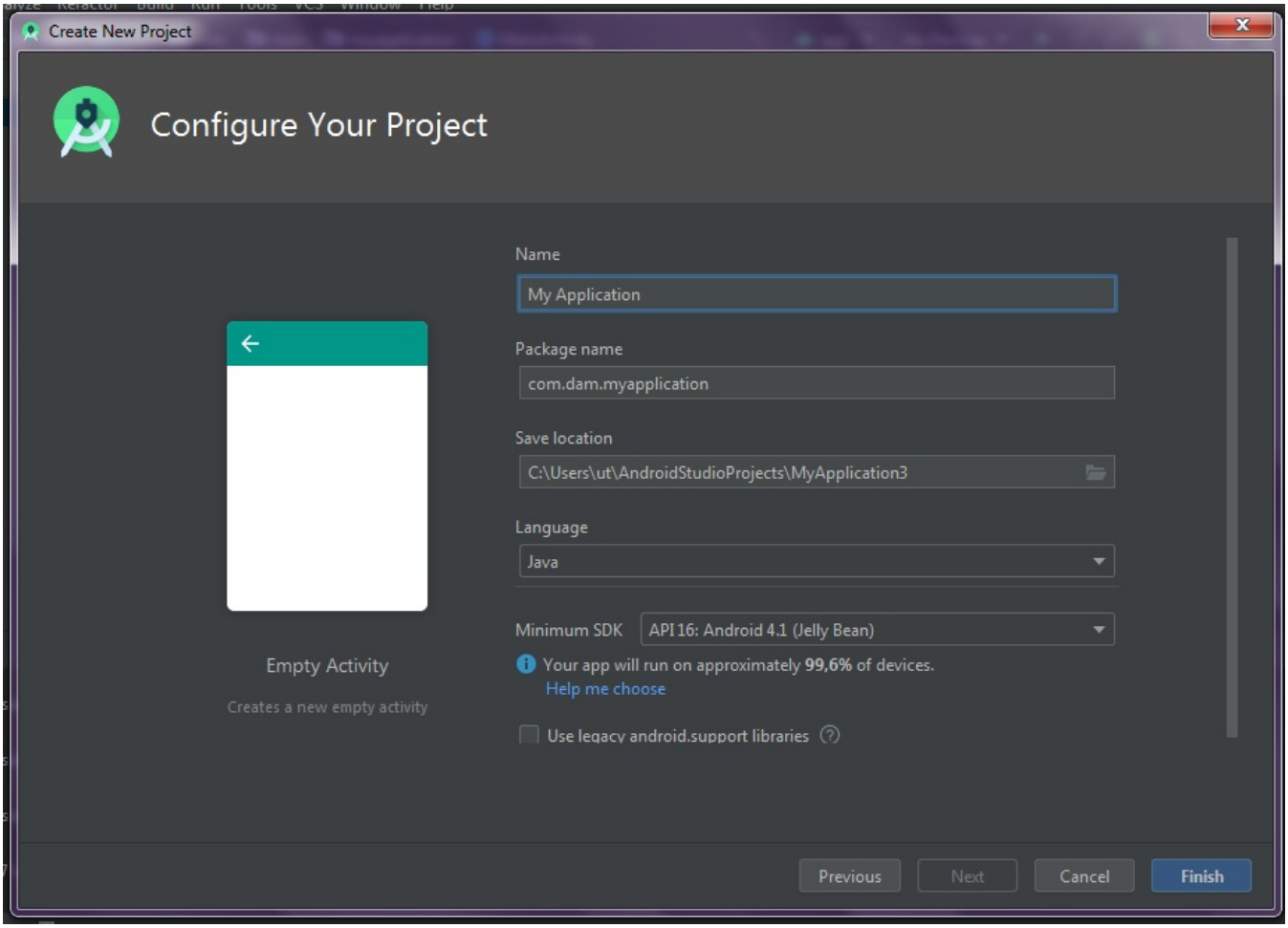


- L'assistant de création d'un nouveau projet démarre.

- Sous **Select a Project Template**, sélectionnez **Empty Activity** et cliquez sur **Next**.



La fenêtre suivante s'affiche ; elle permet de configurer votre projet, suivez les étapes qui sont décrites dans la page suivante :



Remplir les différents champs

- **Application name :** est le nom de l'application qui apparaît aux utilisateurs (en particulier sur le Play Store)**.**
- **Package name :** fournit un qualificatif qui sera ajouté au nom du paquet ; Androïde studio se souviendra de ce qualificatif pour chaque nouveau projet que vous créez.
- **Package Name** : est le nom complet du projet (en suivant les mêmes règles que celles pour nommer les paquets dans le langage de programmation Java). Votre nom de package doit être unique pour tous les paquets installés sur le système Androïde. Vous pouvez **Modifier** cette valeur indépendamment du nom de l'application ou le domaine de l'entreprise.
  - **Save location :** est le répertoire de votre système qui contient les fichiers de projet.
- **Minimum SDK** : L'API minimale définit la version minimale de l'API d'Android avec laquelle votre application doit être compatible. Plus que cette valeur est petite plus que l'app cible plus de marché, cependant, vous perdez quelques fonctionnalités nouvelles supportées par les nouvelles versions.

  Sélectionner API 16 (Android 4.1, Jelly Bean)

- Cliquez sur **Next.**
- Sous **Configure the activity**, changer le **Activity name** à "*MyActivity*". Le **layoutname** passe à "*activity_my*".
- Cliquez sur le bouton **Finish** pour créer le projet.

Votre projet Android est maintenant une application de base qui contient des fichiers par défaut dont la structure est la suivante :

- app / src / main / res / layout / activity_my.xml

  Ce fichier représente le layout XML représentant l'interface écran de l'activité my_activity. Il contient certains paramètres et un contrôle TextView affichant par défaut le message "Hello World !".

- app / src / main / java / com.mycompany.myfirstapp / MyActivity.java

  Ce fichier est créé automatiquement par l'assistant, il représente le code associé à l'interface précédente. Il contient une classe nommée « MyActivity » qui étend la classe AppCompatActivity de la plateforme. Son rôle principal est de charger le layout associé et de gérer les différents contrôles. Remarquer que cette activité va-t-être lancée automatiquement (par défaut).

- app / src / main / AndroidManifest.xml
  Ce fichier décrit les caractéristiques (configuration) fondamentales de l'application.

- app / build.gradle

  Android Studio utilise par défaut Gradle pour compiler et construire votre application. Il y a un fichier build.gradle pour chaque module de votre projet, ainsi qu'un dossier build.gradle pour l'ensemble du projet. Habituellement, vous êtes seulement intéressé par le fichier build.gradle de chaque module d'application. Ceci est où les dépendances de construction de votre application sont définies, y compris les paramètres par défaut tels que :
  - compiledSdkVersion est la version de la plate-forme sur laquelle vous compilez votre application.
  - applicationId est le nom du package complet pour votre application.
  - minSdkVersion est la version SDK minimale supportée par l'app.
  - targetSdkVersion indique la plus haute version d'Android avec laquelle vous allez tester votre application.

- `apply plugin: 'com.android.application'`

```
android {
    compileSdkVersion 29
    buildToolsVersion "29.0.3"

    defaultConfig {
        applicationId "com.dam.tp02"
        minSdkVersion 21
        targetSdkVersion 29
        versionCode 1
```

```
            versionName "1.0"

            testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
        }

        buildTypes {
            release {
                minifyEnabled false
                proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'),
    'proguard-rules.pro'
            }
        }

    }
    dependencies {
        implementation fileTree(dir: 'libs', include: ['*.jar'])

        implementation 'androidx.appcompat:appcompat:1.1.0'
        implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
        testImplementation 'junit:junit:4.12'
        androidTestImplementation 'androidx.test.ext:junit:1.1.1'
        androidTestImplementation 'androidx.test.espresso:espresso-core:3.2.0'
    }
```

Notez également les sous-répertoires contenus dans le répertoire /res, ils définissent l'ensemble des ressources de votre application, en particulier :

- drawable-<*density*>/ : Répertoires pour les ressources images, autres que les icônes de lancement, conçus pour différentes tailles d'écran.
- layout/ : Répertoire pour les fichiers qui définissent l'interface de votre application utilisateur comme activity_my.xml.
- menu/ : Répertoire des fichiers qui définissent les objets menus de votre application.
- mipmap / : Icônes de lancement de l'app par taille. C'est cette icône qui va se placer dans la liste des apps (Home) permettant de lancer l'app.
- valeurs/ : Répertoire contenant d'autres fichiers XML représentant une collection de ressources, tels que chaînes de caractères, couleurs et autres.

- Une application contient un grand nombre de fichiers. L'onglet "Projets" de l'éditeur vous permet d'accéder à ces fichiers. Cet onglet propose différents modes :
  - Le mode Androïde (par défaut) : structure ces fichiers dans des sous-dossiers logiques pour faciliter vos tâches de développement.
  - Le mode Projet : vous permet quant à lui de voir l'arborescence réelle de vos fichiers sur votre disque.

•Pour compiler une application, Androïde se sert d'un outil nommé Gradle qui permet d'effectuer toutes les tâches complexes nécessaires à la compilation.

2. Exécution de l'application

- Pour exécuter l'application Androïde que vous développez, vous avez 2 options :
  - Utiliser un téléphone Androïde réel si vous en avez un à disposition.
  - Utiliser un téléphone virtuel via l'Androïde Virtual Device (AVD) ou l'émulateur ;

2.1 Installer un appareil mobile Androïde

1. Branchez votre appareil à votre machine de développement avec un câble USB. Vous aurez besoin, éventuellement d'installer le driver approprié (généralement un driver standard ADB driver).
2. Activer le **débogage USB** sur votre appareil. Sur Androïde 4.0 et plus récent, allez à **Paramètres> Options pour les développeurs**.

2.2 Exécutez l'application à partir d'Androïde studio

1. Sélectionnez l'un des fichiers de votre projet et cliquez sur **Exécuter** à partir de la barre d'outils ou tapez SHIFT+F10.
2. Dans l'option **Choose Device window** de la fenêtre qui apparaît, sélectionnez **running device**, puis sélectionner votre appareil. Cliquer OK.

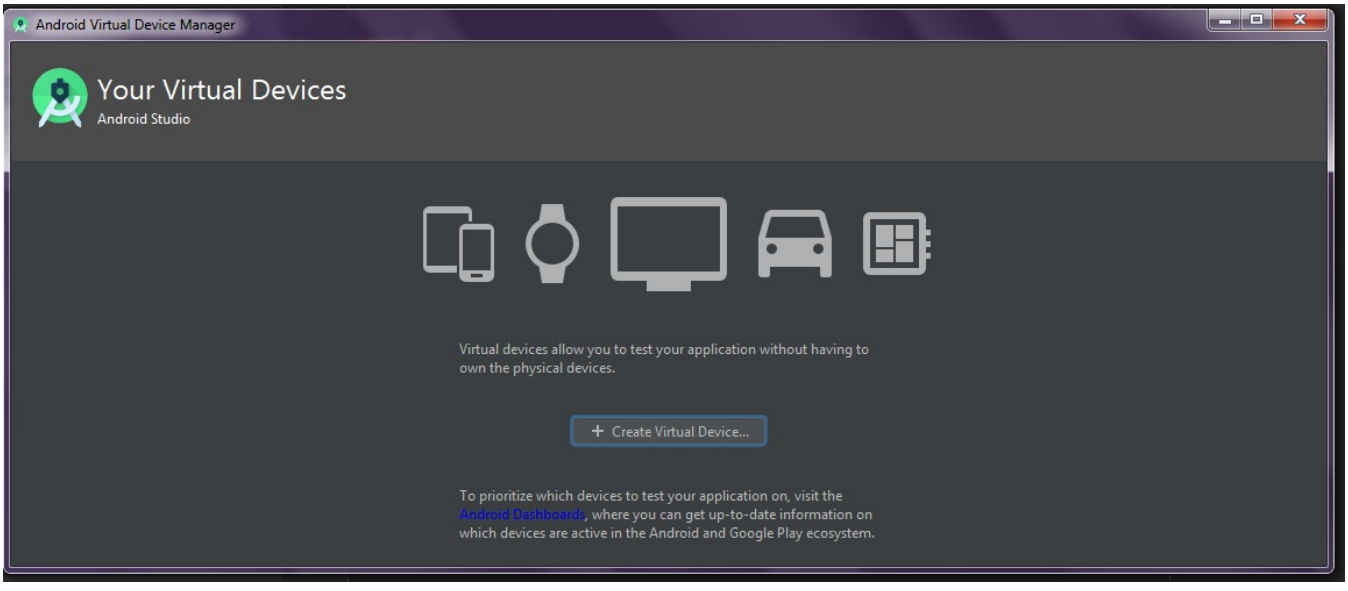
Androïde studio installe l'application sur votre appareil connecté et la démarre.

2.3 Exécuter sur un émulateur AVD

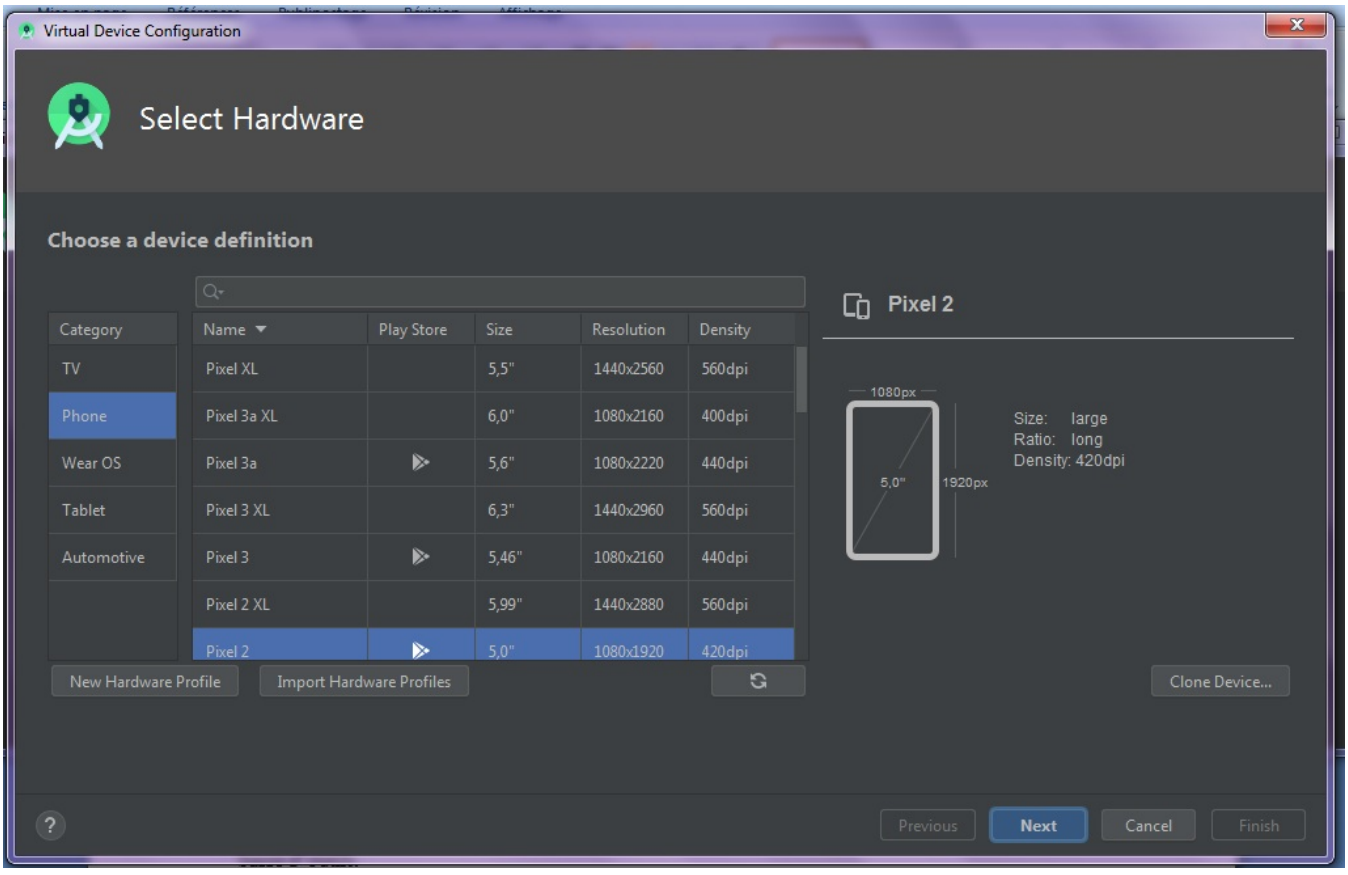
Un émulateur Androïde, AVD, est une configuration logicielle permettant de modéliser un dispositif matérielle spécifique tournant sous Androïde.

2.3.1 Créer un AVD

1. Lancez **l'Androïde Virtual Device Manager** :



- Dans Androïde Studio, sélectionnez **Tools> Android> AVD Manager**, ou cliquez sur l'icône AVD Manager  dans la barre d'outils. L'AVD Manager apparaît (voir la figure ci dessous).
- Sur l'écran principal AVD Manager, cliquez sur Create Virtual Device.
- Dans la fenêtre **Material selection**, sélectionnez une configuration de l'appareil, comme Nexus 6, puis cliquez sur **NEXT**.
- Sélectionnez la version désirée du système pour l'AVD et cliquez sur **NEXT**.
- Vérifiez les paramètres de configuration, puis cliquez sur **FINISH**.



- Dans **Choose Device definition**, cliquez sur **choisir l'option**
- Sélectionnez l'émulateur que vous avez créé, puis cliquez sur N**ext**.
- Il peut prendre quelques minutes pour que l'émulateur se charge. Enfin, l'App apparaît sur l'écran de l'émulateur.