

# *Découverte de l'environnement de développement Android Studio*

## *Présentation d'Android Studio*

Android Studio est l'environnement de développement spécifique à Android. Il est basé sur **IntelliJ IDEA**, autre environnement connu des développeurs Java.

Android Studio peut être téléchargé à partir du site *[developer.android.com](https://developer.android.com)*.

Android Studio utilise **Gradle** pour automatiser la construction et la gestion de projets (Gradle fournit l'équivalent des outils Maven et Ant). Gradle permet de configurer le processus de compilation de l'application, créer des fichiers APK selon la configuration de projet souhaitée, gérer les dépendances et les bibliothèques utilisées dans le projet ...

Android Studio intègre un utilitaire nommé **SDK Manager** permettant de gérer les différentes versions d'Android pour lesquelles il est possible de développer.

## **SDK Android**

Le SDK (Software Development Kit) contient tous les outils nécessaires pour créer une application Android. Ces principaux constituants sont les outils suivants :

- **AAPT** - Android Asset Packaging Tool  
Cet outil sert à créer et analyser des fichiers \*.apk
- **ABD** - Android Debug Bridge  
Le but de cet outil est d'établir et de gérer la connexion à un terminal Android ou à un émulateur. Il sert aussi à transférer une application ou des fichiers vers le terminal ou l'émulateur
- **DDMS** - Dalvik Debug Monitor Service  
Cet outil est utilisé pour le débogage de l'application et permet de faire des captures d'écran, voir les *threads* en cours d'exécution, voir les messages de log, connaître la liste des processus en cours d'exécution, simuler l'envoi de messages et d'appels, simuler une localisation ...
- Le SDK contient aussi un utilitaire permettant de créer et de gérer les **émulateurs** ainsi que la documentation pour chaque version d'Android et des exemples pour les différentes API.

## Complément à propos du SDK Manager

Lors de l'installation d'Android Studio, seuls les éléments de base indispensables au fonctionnement de l'environnement sont installés. En particulier, une seule version du SDK est installée. Il faut ensuite télécharger des composants supplémentaires.

Android Studio comporte un outil nommé **Android SDK Manager** qui permet de gérer les composants spécifiques aux versions de plate-forme Android ciblées, c'est-à-dire celles pour lesquelles on souhaite développer des applications.

Pour accéder au SDK Manager, aller sur le menu « Tools », sous-rubrique « SDK Manager ». On peut aussi passer par la rubrique « Settings ... » du menu « File », puis rubrique « Languages Frameworks ».

En bas de la fenêtre qui s'affiche, il peut être intéressant de cocher *Show Package Details*.

L'onglet *SDK Platforms* permet de télécharger et d'installer les packages qui correspondent aux API pour lesquelles on souhaite développer, ou bien les images systèmes que l'on souhaite installer sur un émulateur.

L'onglet *SDK Tools* permet de télécharger et d'installer différents outils tels que les outils de Build, ceux permettant de créer et gérer des émulateurs ou l'utilitaire HAXM qui permet d'accélérer le fonctionnement d'un émulateur sur un ordinateur doté d'un processeur Intel.

## Exercice 1- Premier projet

1) Créer un nouveau projet ayant pour nom : AfficheBasique.

Pour ce faire : File | New | New project

Ou bien dès le lancement d'Android Studio, celui-ci va vous proposer de créer un nouveau projet.

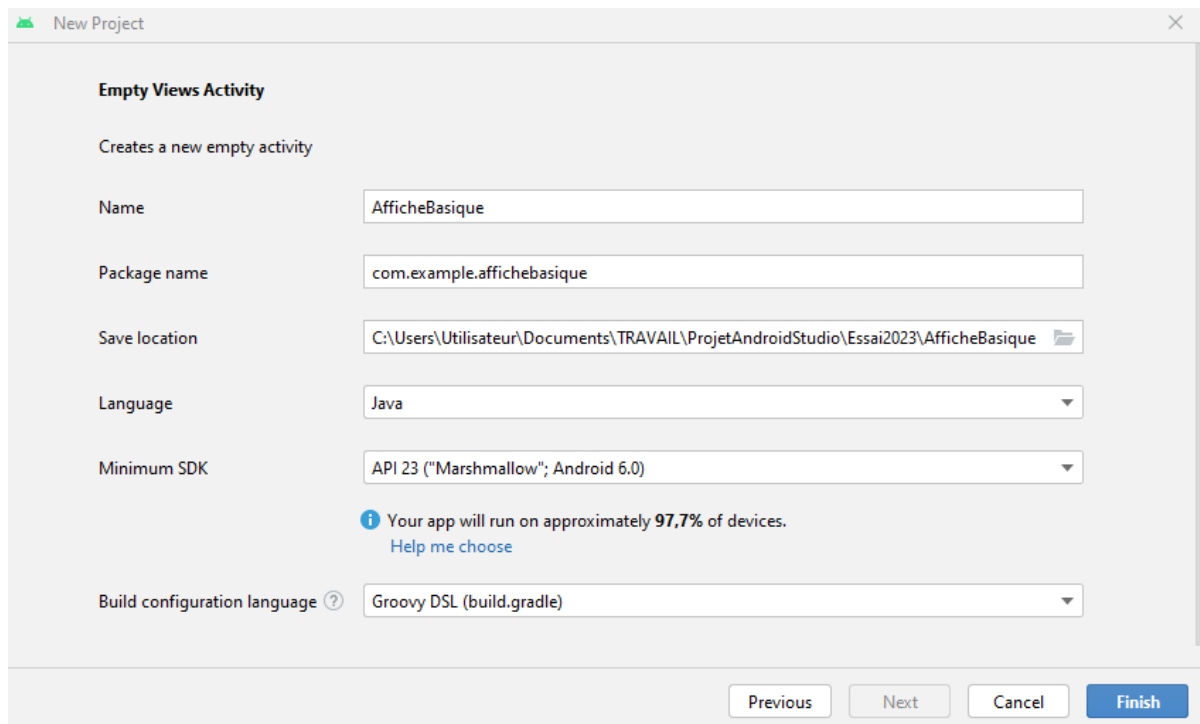
La fenêtre suivante nous propose de choisir un type d'activité pour notre application. Il faut choisir **Empty Views Activity**, puis cliquer sur *Next*.

Dans la fenêtre qui s'ouvre, compléter les rubriques de manière à indiquer :

- le nom de l'application
- le nom du paquetage contenant l'application
- l'emplacement dans lequel le projet sera stocké. Le mieux est de créer sur votre compte un *workspace* spécifique qui contiendra tous vos projets Android.

Dans la copie d'écran :

- le *workspace* se nomme *ProjetsAndroidStudio*. Mais vous pouvez lui attribuer le nom de votre choix.
- Le nom du projet est *AfficheBasique*.
- Pour la version du SDK, choisissez **API 23 : Android 6.0 (Marshmallow)**. Il s'agit de la version d'Android minimale pour laquelle l'application que l'on crée va pouvoir fonctionner (elle fonctionnera sur tous les terminaux ayant au moins la version Marshmallow d'Android)
- Pour le choix du langage, par défaut c'est Kotlin qui est positionné, veuillez bien à choisir **Java**
- Le langage de configuration du *build* : choisir « Groovy DSL (build.gradle). Groovy est le langage par défaut des fichiers *gradle*.



Cliquer sur *Finish*.

Après quelques secondes, le projet sera créé. Remarquez qu'Android Studio affiche tout en bas à droite, les tâches en cours de réalisation (importation diverses, téléchargement d'une version spécifique de Gradle, indexation ...)

Ensuite, toujours en bas de la fenêtre vous verrez le message «*indexing*». L'étape de création du projet est presque terminée. Dès lors qu'aucun message n'est affiché en bas de la fenêtre, le projet est créé.

## Exercice 2 : Créer un émulateur

L'émulateur est un outil permettant de simuler le fonctionnement d'un téléphone ou d'une tablette Android. Il permet de tester les applications Android sans avoir besoin d'un véritable appareil physique, et aussi de tester le bon fonctionnement sur des configurations de terminaux dont nous ne disposons pas.

Il est possible d'utiliser aussi des outils gérant les émulateurs et indépendants d'Android Studio. Par exemple, **Genymotion** est une alternative. Il permet de créer et d'utiliser des émulateurs au sein d'une machine virtuelle, ce qui augmente les performances de ceux-ci.

Pour utiliser Genymotion, il faut d'abord télécharger et installer l'outil *VirtualBox* (site [www.virtualbox.org](http://www.virtualbox.org)). L'utilisation de Genymotion est gratuite pour un usage personnel et nécessite la création d'un compte sur le site [www.genymotion.com](http://www.genymotion.com).

## AVD Manager d'Android Studio

Le gestionnaire des émulateurs permet de paramétrer un ou plusieurs terminaux mobiles (téléphone, tablette ...). Chaque configuration est stockée dans un AVD (Android Virtual Device), et est créée via l'AVD Manager. Une fois qu'Android Studio est ouvert, pour lancer l'**AVD Manager**, il faut cliquer sur



l'icône de la barre d'outil, ou bien passer par le menu **TOOLS | Device Manager**.

## Autoriser la virtualisation sur votre ordinateur

Eventuellement, sur la fenêtre « Your Virtual Devices », vous verrez un message sur fond jaune « VT-x disabled in BIOS » signifiant que vos paramètres systèmes n'autorisent pas la virtualisation. Celle-ci est nécessaire pour

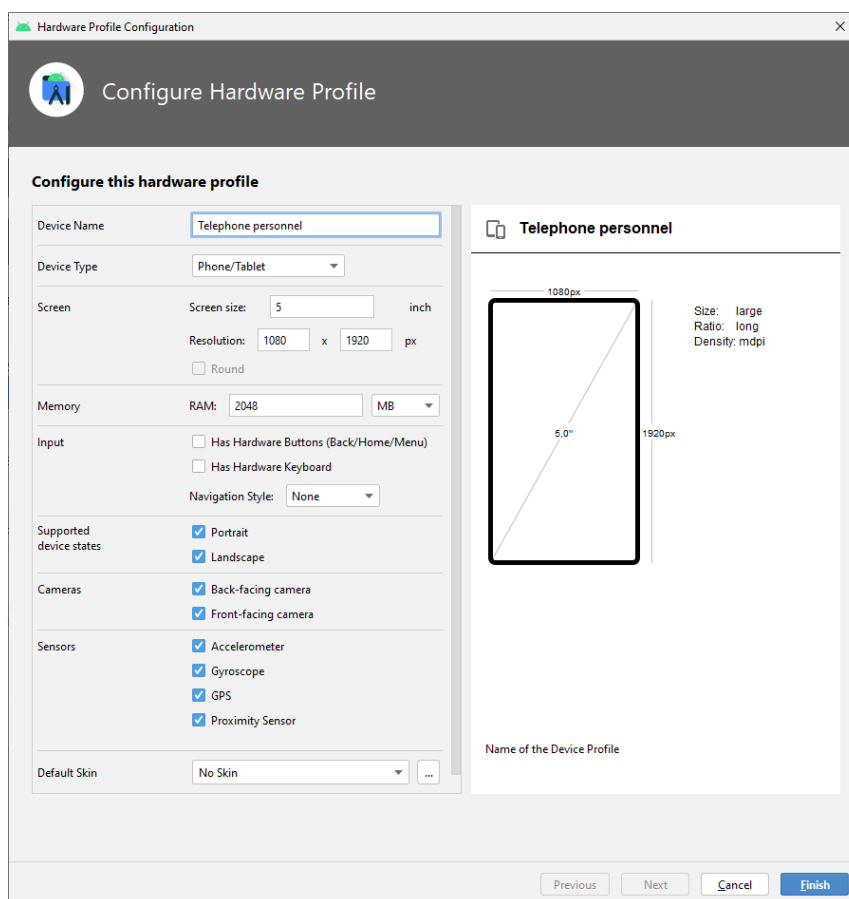
## Création d'un AVD

Une vue permettant de créer et gérer des émulateurs s'affiche :



Cliquer sur « Create device » pour créer un nouvel émulateur. Puis sur la fenêtre qui s'ouvre et qui s'intitule « Select Hardware », cliquer sur « New Hardware Profile » pour créer un nouvel AVD.

Une fenêtre intitulée *Configure Hardware Profile* s'ouvre.



On peut déjà renseigner le nom à attribuer au téléphone et une partie de ses caractéristiques. En fait, il s'agit ici des caractéristiques matérielles.

Au niveau de la RAM du téléphone virtuel, 2 options sont possibles :

- si votre ordinateur dispose de 8 Go de RAM, vous pouvez laisser la valeur par défaut 2048 MB
- si par contre votre ordinateur est doté de seulement 4 Go de RAM, il est opportun de diminuer la RAM du téléphone à 1024 MB.

Concernant la résolution de l'écran du téléphone virtuel, la valeur par défaut est 1080 x 1920 px. Vérifiez que l'écran de votre ordinateur propose au moins cette résolution. Sinon, il faut diminuer la résolution du téléphone.

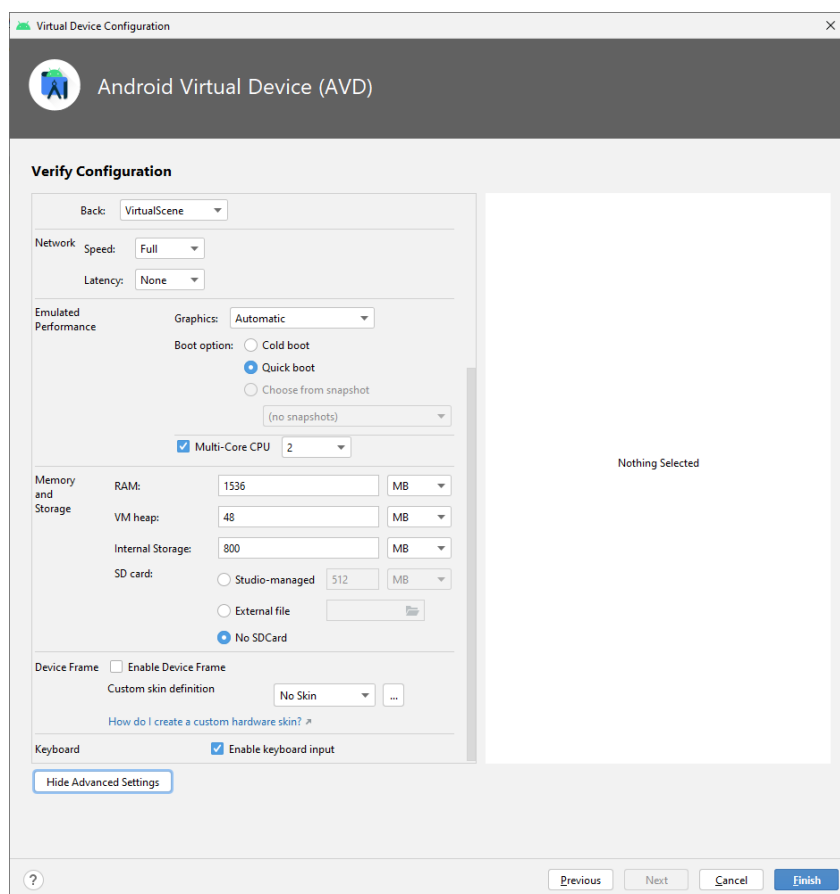
On clique ensuite sur *Finish*.

Le nouvel AVD apparaît alors dans la liste des AVD disponibles. Par défaut, il est sélectionné. Il faut encore indiquer la version d'Android qui sera installée sur cet AVD. Pour ce faire, cliquer sur *Next*.

Sur la fenêtre suivante, il faut indiquer l'image système que l'on souhaite utiliser. Nous allons choisir une image de la catégorie *x86 Images* (sélectionner le 2<sup>ème</sup> onglet), avec la version *Marshmallow* correspondant à *Android 6.0 (with Google APIs)*. Cliquer d'abord sur le lien « Download » afin de télécharger cette image.

Une fois le téléchargement terminé, cliquer sur *Next*.

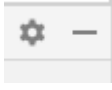
Vous pouvez maintenant vérifier les paramètres de l'AVD, et éventuellement les modifier pour qu'ils soient conformes à ceux que l'on souhaite. Cliquer sur *Show Advanced Settings* pour vérifier les autres paramètres. Et ensuite sur *Finish*.

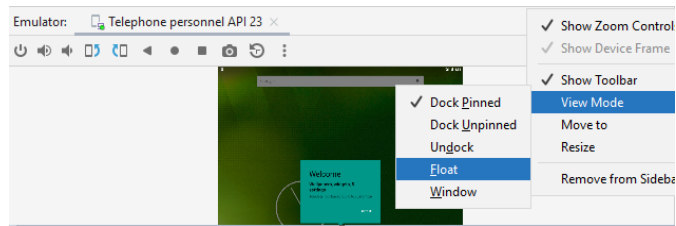



### **Exercice 3 - Lancer le premier émulateur et exécuter l'application**

L'émulateur créé apparaît maintenant dans le panneau de gauche, celle du « Device manager ». Cliquer sur la flèche grise située à droite de l'émulateur que vous avez créé.



L'émulateur s'affichera en dessous de la vue « Device manager ». Si l'on souhaite dissocier son affichage d'Android Studio, on peut cliquer sur l'icône d'ouverture du menu , puis sélectionner « View Mode », puis « Float ».



Si la vue du *Device Manager* ne nous est plus utile, on peut la réduire en cliquant sur .

Exécuter l'application sur l'émulateur préalablement lancé en faisant **Run** | **Run 'app'** ou bien cliquer sur la flèche verte dans la barre d'outils.

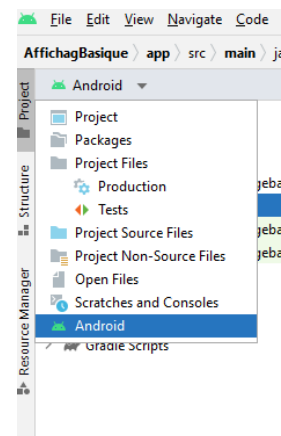
Vérifier le bon fonctionnement de votre application.

## Exercice 4- Arborescence du projet

Le but de l'exercice est de se familiariser avec l'arborescence du projet, en répondant aux questions suivantes.

### Remarque

Il est possible de visualiser l'arborescence des fichiers constituant le projet de plusieurs manières différentes. Pour ce faire, au dessus de la vue montrant cette arborescence, dans la liste déroulante, on peut choisir un type de visualisation. Par défaut, c'est le mode « *Android* » qui est actif. C'est le mode le plus pratique. Mais on peut aussi choisir « *Project* » ou « *Project Files* », par exemple.



- a) Dans le fichier `AndroidManifest.xml` (voir dossier *manifests*), déterminer :
  - quel est le nom de propriété à travers laquelle est défini le nom de l'application
  - combien d'activités constituent l'application
  - le numéro de la version minimale de l'API Android pour laquelle cette application peut fonctionner (si mentionnée dans ce fichier)
  - le numéro de la version cible de l'API Android pour laquelle cette application est optimisée (si mentionnée dans ce fichier)
- b) Notre application comporte une seule activité. Vérifier que son nom est bien *MainActivity*, dans le fichier `AndroidManifest.xml`. Quel est le nom de la balise dans laquelle on indique toutes les caractéristiques d'une activité ? Supposons qu'une application comporte 3 activités, à votre avis combien de fois trouvera-t-on cette balise, imbriquée dans celle qui décrit l'application ?

- c) Dans le dossier des ressources `res`, retrouver le fichier qui contient l'icône de lancement de l'application (bonhomme vert android).
- d) Dans le dossier des ressources `res`, rechercher le fichier `strings.xml`. Combien de chaînes de caractères définit-il ? Quel est l'identifiant de cette chaîne ou de ces chaînes, et leur contenu ?
- e) Dans le dossier des ressources `res`, retrouver le fichier décrivant la mise en forme de l'activité, c'est-à-dire sa vue. Quel est le nom du gestionnaire de mise en forme utilisé pour cette activité ?
- f) Trouver le fichier `.java` qui contient l'activité principale de l'application. Combien de méthodes contient la classe activité ? Indiquer le nom de la méthode ou des méthodes.
- g) Il existe plusieurs fichiers `build.gradle`, l'un pour le projet, l'autre pour l'unique module présent dans le projet. Est-ce le fichier `build.gradle` du niveau module ou du niveau projet qui contient la version minimale d'Android sur laquelle notre application peut fonctionner ?
- h) Avec quelle version du SDK le projet va-t-il être compilé ?
- i) Faire en sorte que le texte affiché soit en gras, en couleur (à vous de choisir laquelle), et plus grand qu'initialement (pour ce faire, voir le cours page 42).

## **Exercice 5- Modification du projet**

La chaîne de caractères affichée par le projet créé par défaut « Hello world ! » n'est pas définie en tant que constante parmi les ressources de l'application. Cette façon de faire n'est pas optimale. Vous allez améliorer ce point.

- a) Définir dans le fichier `strings.xml`, une nouvelle chaîne de caractères ayant pour valeur « Hello world ! ». Vous pouvez lui associer l'identifiant de votre choix.
- b) Modifier le fichier ***layout activity\_main.xml*** pour que la chaîne affichée ne soit pas écrite en dur. A la place, il faut utiliser l'identifiant que vous avez choisi à la question précédente (voir la syntaxe dans l'exemple du cours)
- c) Réinstaller l'application sur l'émulateur pour vérifier le bon fonctionnement.
- d) Modifier la chaîne affichée afin qu'elle contienne une apostrophe (voir la syntaxe dans le cours), par exemple « Bonjour de la part des étudiants de l'IUT de Rodez ».
- e) Vérifier le bon fonctionnement
- f) Faire en sorte que le texte affiché soit en gras, en couleur (à vous de choisir laquelle), et plus grand qu'initialement (pour ce faire, voir le cours).
- g) Vérifier le bon fonctionnement

- h) Par défaut, il est possible que la barre d'action ne s'affiche pas en haut de l'écran du terminal. Pour la faire apparaître il faut modifier le thème de l'application. Pour ce faire, dans le fichier *themes.xml*, il faut remplacer le thème *AppTheme* par

```
<style name="Base.Theme.AfficheBasique" parent="Theme.AppCompat.Light.DarkActionBar">
    <!-- Customize your light theme here. -->
    <!-- <item name="colorPrimary">@color/my_light_primary</item> -->
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent</item>
</style>
```

Ce thème fait référence à 3 couleurs qui ne sont pas définies par défaut. Il faut donc les ajouter au fichier des couleurs. Ce fichier est créé par défaut avec le nom *colors.xml*. On doit lui ajouter :

```
<color name="colorPrimary">#3F51B5</color>
<color name="colorPrimaryDark">#303F9F</color>
<color name="colorAccent">#FF4081</color>
```

- i) Vérifier le bon fonctionnement

## Exercice 6- Recherche d'informations sur les versions d'Android

Pour obtenir des informations sur les proportions des versions d'Android, il faut commencer par les mêmes étapes que pour la création d'un nouveau projet :

File | New | New project ...

Choisir « Empty Views Activity », puis cliquer sur « Next ».

Sur la fenêtre de création du projet, cliquer sur [Help me choose](#)

- Quelle est la proportion des terminaux qui utilisent la version 13 d'Android ?
- Quelle est la proportion des terminaux qui utilisent la version 11 ?
- Si l'on choisit Marshmallow comme version minimale d'Android pour notre application, sur quel pourcentage des terminaux pourra-t-on installer l'application ?
- Même question avec Pie.
- Quelle est la version la plus courante ? Est-ce la dernière version ?
- Quelle est la 2<sup>ème</sup> version la plus courante ? Est-ce la dernière version ?
- A chaque version d'Android, un seul niveau d'API est associé. Cette affirmation est-elle vraie ou fausse ?

## Exercice 7 - Recherche d'informations sur le site de référence

Un site Internet est dédié au développement sous Android. Le but de cet exercice est de découvrir son contenu : <http://developer.android.com>

Les informations les plus utiles pour le développement d'applications sont disponibles avec les URL suivantes :

**Guides** : <http://developer.android.com/guide>

**Reference** : <http://developer.android.com/references>



**Android Studio** : <http://developer.android.com/studio>

**Partie reference :**      Overview      Guides      UI Guides      Reference      Samples

Overview	Liste des principales notions détaillées dans les autres rubriques de l'option <i>Docs</i>
Guides	Documentation sur les principaux concepts
UI Guides	Documentation sur les concepts plus avancés en lien avec les interfaces
Reference	manuel de référence de l'API, tous les composants sont expliqués, classés par package
Samples	des exemples

### Les informations demandées sont à rechercher sur le site.

1. Pour localiser les réponses à cette question, le mot-clé à rechercher est **dashboard**. Quel type d'écran est le plus représenté (en combinant taille et densité) ? Quel est le deuxième type d'écran le plus représenté ?
2. OpenGL est une API de programmation graphique 2D ou 3D disponible sous Android (et aussi sur de nombreuses autres plateformes). Quel est le numéro de la dernière version d'OpenGL et quel est la proportion des appareils mobiles qui intègrent cette version ?
3. Dans la rubrique Guides, sous rubrique User location, trouver les noms des 2 permissions qu'il faut éventuellement déclarer dans le fichier *Manifest* si l'on souhaite que notre application accède à la localisation du terminal.
4. Parcourir les rubriques Reference et Android Platform. Déterminer si les packages usuels du langage Java sont bien présents dans les bibliothèques Android : *java.util*, *java.math*, *java.io*, *java.util.regex*, *javax.swing* ...
5. Dans la rubrique Android Studio, trouver la quantité de RAM minimale et aussi l'espace disque nécessaire pour installer Android Studio (sous Windows), le SDK d'Android et le gestionnaire des émulateur).
6. Consulter la sous-rubrique User Guide puis Meet Android Studio dans le but de trouver :
  - le nom de l'IDE à partir duquel Android Studio a été conçu
  - 5 raccourcis (*shortcut*) que pourraient être utiles pour l'utilisation courante d'Android Studio (cliquer aussi sur le lien *Keyboard shortcuts* pour une liste complète)

- Quelques informations (à résumer en 2 ou 3 phrases) à propos du SDK Manager. Pour ce faire, voir la sous-rubrique *Update the IDE and Tools*.

7. Rechercher des informations sur la notion d'ActionBar. Résumer celle-ci avec 4 phrases pertinentes.

8. Quelle est la différence entre un style et un thème ? Rechercher des explications sur ces 2 notions. Ecrire 3 phrases pertinentes qui donneraient des informations pertinentes pour chacune des deux notions. Quelle est la différence entre Theme.Holo et Theme.Holo.Light ?

9. Que permet de faire l'outil lint ?

10. Qu'est que SQLite ? Résumer en 2 ou 3 phrases pertinentes.

11. Rechercher la page décrivant le widget **TextView**. Tous les *widgets* sont à peu près décrits de la même manière, avec les mêmes rubriques. Quelles sont les principales rubriques décrivant un *widget TextView* ? Vous devez trouver, environ, de 5 à 8 rubriques, selon que vous regroupez ou pas certaines d'entre elles.

12. Qu'est-ce que **Material Design** ? Résumer ce concept en quelques phrases.

13. Si une application a besoin d'une connexion Internet, il faut ajouter une permission dans le fichier *manifest*. Comment écrit-on précisément cette permission ?

14. Qu'est-ce que **Kotlin** ? Résumer en 2 ou 3 phrases.