1. PRÉPARATION DE L'ENVIRONNEMENT DE TRAVAIL

Etape 1 : Installation du JDK

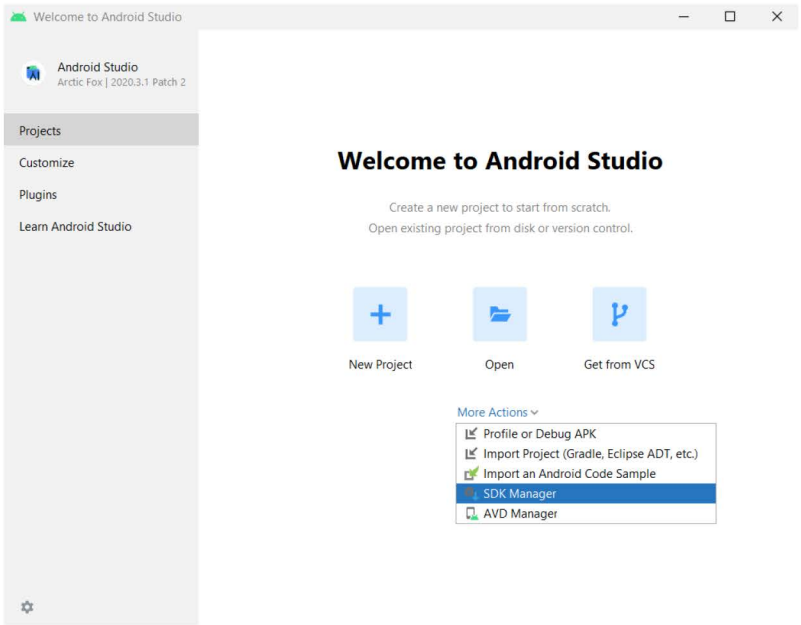
Le développement des applications Android ainsi que les bibliothèques utilisées sont basés sur le langage Java. Donc, le premier composant qui doit être installé est l'outil de développement Java JDK.
JDK peut être téléchargé du site officiel d'Oracle à l'adresse suivante :
http://www.oracle.com/technetwork/java/javase/downloads/index.html
NB : Il est recommandé de télécharger la dernière version du JDK.

Remarque JRE : Avant d'installer un environnement de développement, on doit disposer un environnement d'exécution Java ou JRE (pour *Java Runtime Environment)* installé et correctement configuré sur la machine.

Etape 2 : Installation d'Android Studio et configuration du SDK

Android Studio est disponible à l'adresse suivante :
https://developer.android.com/studio/index.html

A cette adresse, vous pouvez télécharger l'IDE Android Studio ainsi que son SDK chacun à part. Cela fait, il est nécessaire de paramétrer Android Studio afin de pointer vers le répertoire de SDK en allant : More Actions→SDK Manager, cliquez ensuite sur Edit de l'option Android SDK Location afin de sélectionner le chemin de SDK..



Etape 3 : Utilisation du SDK Manager

Dans le SDK Manager, cocher la case « Show Package Details » et vérifier que les packages suivants sont installés :

- Android > Android SDK Plateform

Dans l'onglet SDK Tools, assurez-vous que les packages suivants sont installés

- Android SDK Build-Tools
- Android Emulator
- Android SDK Platform-tools
- Google Play services
- Intel x86 Emulator Accelerator (HAXM installer)

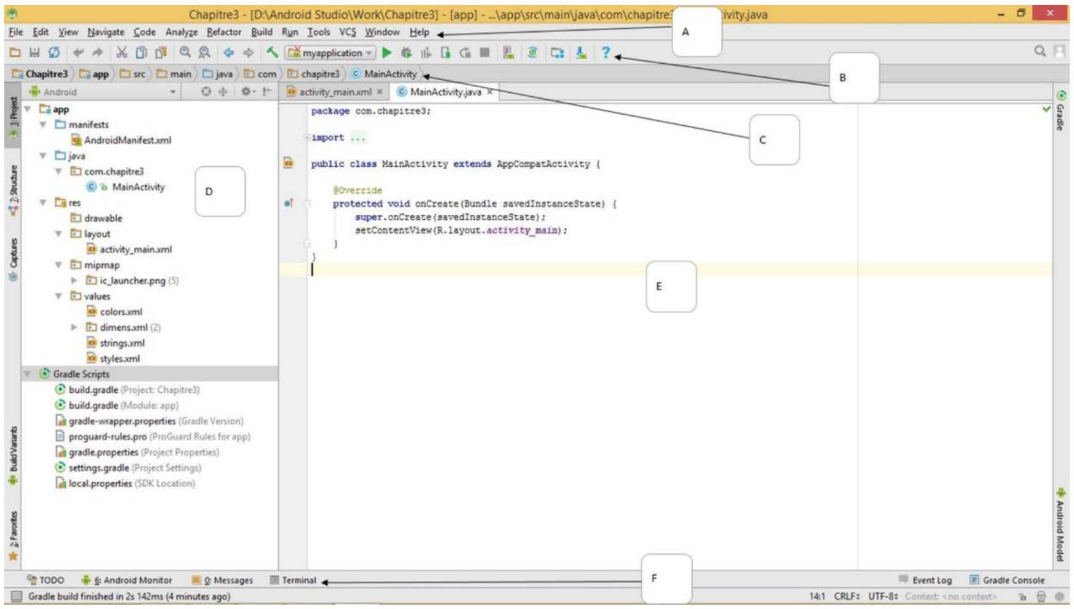
2. MA PREMIÈRE APPLICATION ANDROID

L'objectif de cet atelier est d'expliquer les différentes étapes à suivre pour programmer une activité Android. L'activité à programmer contient un seul bouton « Afficher », lorsque l'utilisateur clique sur ce bouton, le message « **Ma première activité Android** » s'affiche.

Etape 1 : Création du projet

1. Cliquer sur « New Project ». L'assistant affiche une interface de configuration du nouveau projet
2. Rester dans l'onglet « Phone and Tablet » et définir le type de l'activité de l'application : Choisir « Empty activity »
3. Entrer le nom de l'application dans le champ correspondant, le package (ex : com.affiche), choisir l'emplacement, le langage de développement (Java) et enfin une valeur pour le SDK minimal : Choisir l'API 21 par exemple
4. Cliquer sur « Finish » pour créer le projet.

Etape 2 : Description de l'interface



Cette interface est décomposée en un ensemble d'éléments comme suit :

- A : barre des menus pour diverses tâches en rapport avec l'environnement d'Android Studio.
- B : barre des outils représentant des raccourcis vers les tâches les plus fréquentes.
- C : barre de navigation permettant de naviguer à travers les répertoires.
- D : fenêtre du projet qui permet d'afficher tous les fichiers du projet organisés selon une vue hiérarchique. La vue par défaut étant « Project », il en existe 8 différentes dont la vue « Android » est conçue pour améliorer la gestion des fichiers des projets Android.
- E : fenêtre d'édition pour éditer les fichiers du projet.
- F : barre des états d'affichant divers informations telles que : l'état du projet, la mémoire utilisée, le journal

des erreurs, ...

Etape 3 : Description de l'arborescence

L'arborescente du projet Android est décrite selon la vue « Android ». Elle est organisée en trois dossiers contenus dans le dossier portant le nom du module créée par défaut « app » :

- **manifests :** Il contient un fichier XML nommé « AndroidManifest.xml » décrivant les caractéristiques fondamentales de l'application : le nom du package, l'icône, les activités, les permissions,…
- **java :** il englobe les classes Java de l'application. La classe créée par défaut est nommée « MainActivity.java ». C'est une classe de type activité qui hérite de la super classe « Activity »
- **res :** tout ce qui touche à l'interface utilisateur sera intégré dans les sous-dossiers de res, dont voici une brève description :
    - **drawable :** contient tout élément qui peut être dessiné sur l'écran : images, formes, animations, transitions,… Le développeur peut créer jusqu'à 5 dossiers drawable afin de proposer des éléments graphiques pour tout genre d'appareil Android en fonction de sa résolution :
        - **ldpi** pour des écrans de basse résolution (~120dpi).
        - **mdpi** pour des écrans de moyenne resolution (~160dpi).
        - **hdpi** pour des écrans de haute résolution (~240dpi).
        - **xhdpi** pour des écrans ayant une extra haute résolution (~320dpi).
        - **xxhdpi** pour des écrans ayant une extra extra haute résolution (~480dpi).
    - **layout :** regroupe les fichiers XML qui définissent la disposition des composants sur l'écran. Il contient déjà, dès la création du projet, le layout de l'activité principale que nous avons créée, nommé « activity_main.xml »
    - **mipmap :** contient les images de l'icône de votre application sous différentes résolutions.
    - **values :** contient les fichiers XML qui définissent des valeurs constantes :
        - des chaines de caractères : « strings.xml »
        - des dimensions : « dims.xml »
        - des couleurs : « colors.xml »
        - des styles : « styles.xml »
        - …

Etape 4 : Utilisation de l'AVD Manager

Le gestionnaire AVD est intégré dans la plateforme Android SDK et peut être lancé à partir du menu Tools→Android→AVD Manager.

1. Cliquer sur le bouton « Create Virtual Device… »
2. Sélectionner une catégorie de terminal mobile : Choisir « Phone » pour afficher la liste des devices compatibles avec cette catégorie.
3. Sélectionner un device de votre choix (par exemple : Pixel 2)
4. Choisir l'API et l'architecture disponible - télécharger au besoin - (par exemple API 29)
5. Entrer un nom de description pour ce device dans le champ « AVD Name ».
6. Cliquer sur « Finish » pour créer l'AVD.

Etape 5 : Exécution du projet

1. Cliquer sur le triangle vert directement ou bien choisir le menu Run → Run 'app'
2. Choisir l'AVD que vous avez créé et cliquer sur OK.

Etape 6 : Codage et exécution

1. Création de l'interface de l'activité

- Ouvrir le fichier « app/res/layout/activity_main.xml »,

- Double cliquer sur l'onglet « activity_main.xml » pour agrandir l'espace de travail

- Cliquer sur la loupe « Zoom In (+) » pour agrandir l'interface de l'activité
- Cliquer sur le widget « Button » par le bouton gauche, ne pas relâcher, glisser le bouton et le déposer sur l'interface de l'activité

2. Déclaration des constantes chaînes de caractères
- Ouvrir le fichier « app/res/values/strings.xml »
- Ce fichier est écrit avec la syntaxe XML
- Entre <ressources> et </ressources> ajouter la ligne suivante :

```
<string name="afficher">Afficher</string>
```

L'attribut name "afficher" représente le nom de la constante, il sera utilisé par la suite.
"Afficher" qui est dans la balise <string> représente la valeur de la constante.

3. Utilisation de la fenêtre des attributs « Attributes »
- Ouvrir le fichier « app/res/layout/activity_main.xml »
- La vue « Attributes » affiche les propriétés de l'objet graphique sélectionné, elle contient deux colonnes, la première colonne est le nom de la propriété et la deuxième est sa valeur
- Cliquer sur l'interface puis sur « Hello world ! » puis sur « Button » et remarquer que les propriétés changent suivant l'objet graphique sélectionné

4. Suppression de « Hello world ! »
- Ouvrir le fichier « app/res/layout/activity_main.xml »
- Cliquer sur « Hello world ! » et appuyer sur la touche « Suppr » du clavier
- Appuyer sur « Ctrl+Z » pour annuler la suppression
- Dans la vue « Component Tree » cliquer avec le bouton droit sur « textView1 Hello word ! » et cliquer sur « Delete »

5. Modification des propriétés des widgets
- Ouvrir le fichier « app/res/layout/activity_main.xml »
- Cliquer sur le bouton « Button »
- Dans la propriété « Text » taper la valeur « Afficher »
- Ou bien, on déclare une ressource (une constante) dans le fichier « strings.xml » puis cliquer dans la valeur de la propriété « Text » sur le bouton « … » et choisir « afficher », la valeur de « Text » devient « @string/afficher » qui fait référence à la constante déclarée dans « strings.xml »

6. Modification des Ids (Identifiants) des widgets
- Ouvrir le fichier « res/layout/activity_main.xml »
- Cliquer sur le bouton « Afficher »
- Dans la propriété « id », taper la valeur « btnAfficher » puis Entrée.
- Passer à l'onglet Text (code XML de l'interface graphique), @+id/ a été ajouté automatiquement à « btnAfficher » pour demander à android de générer un identifiant à ce bouton, btnAfficher est l'identifiant logique du bouton et il sera utilisé dans les étapes suivantes.

7. Déclaration des attributs
- Ouvrir le fichier « app/java/com.affiche/MainActivity.java »
- Après « public class MainActivity extends Activity {» taper :

```
//attributs
private Button btnAfficher;
```

- Pour faire les importations des packages nécessaires, aller à :
File→Setting →Editor→ General → Auto Import →Java et faire les changements suivants :
  - Modifier « Insert imports on paste » à « Always »
  - Cocher « Add unambigious imports on the fly »

- Cocher « Optimize imports on the fly »

8. Ajout et appel de la méthode initialiser( )

- Pour la méthode « protected void onCreate(Bundle savedInstanceState) {», ajouter à sa fin la méthode init() et choisir « Create method 'ajouterEcouteur() »
- Ajouter le code suivant à la méthode init():

```
private void initialiser() {
    //lier btnAfficher avec le bouton de activity_main.xml
    btnAfficher=findViewById(R.id.btnAfficher);
    ecouteurs() ;
}
```

9. Ajout et appel de la méthode ecouteurs( )

- Cliquer sur l'ampoule devant « ecouteurs() ; »
- Dans « private void ecouteurs() {» taper le code suivant :

```
btnAfficher.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        afficher();
    }
});
```

10. Ajout et appel de la méthode afficher( )

- Cliquer sur l'ampoule devant « afficher(); » et choisir « Create method 'afficher()' » et dans la fenêtre Choose target Class sélectionner « MainActivity »
- Dans « private void afficher() {» taper le code suivant :

```
Toast t = Toast.makeText(getApplicationContext(),
" Ma première activité Android", Toast.LENGTH_LONG);
t.show();
```

- Exécuter l'application et tester le clic sur le bouton « Afficher »