Chapitre 1
Présentation du langage Kotlin

1.1 Définition

Kotlin est un langage de programmation orienté objet, fonctionnel, avec un typage statique qui permet de compiler sur la machine virtuelle Java et JavaScript. Son développement provient principalement d'une équipe de programmeurs chez JetBrains basée à Saint-Pétersbourg en Russie. Le nom de ce langage vient de l'île de Kotline situé en mer Baltique, à 25 km de St. Pétersbourg.

Google annonce pendant la conférence Google I/O 2017 que Kotlin devient le second langage de programmation officiellement pris en charge par Android après Java. Au cours de cette même conférence en 2019, la société Google nous communique que le développement d'applications Android deviendra de plus en plus "Kotlin-first".

C'est un langage :

- à typage statique : le type des variables est défini dans le code, ce qui permet au compilateur de détecter les erreurs de typage avant l'exécution du programme.
- jeune : première version en 2011 et première publication en 2016 sous forme de version stable.
- avec une compilation en byte code : le code source est compilé à l'avance ou à la volée lors de l'exécution dans une représentation intermédiaire, le byte code java.
- avec une exécution JVM : Java Virtual Machine est un appareil informatique fictif qui exécute des programmes compilés sous forme de byte code Java.

1.2 Site officiel

Vous trouverez une mine d'informations sur le site officiel du langage Kotlin :

https://kotlinlang.org

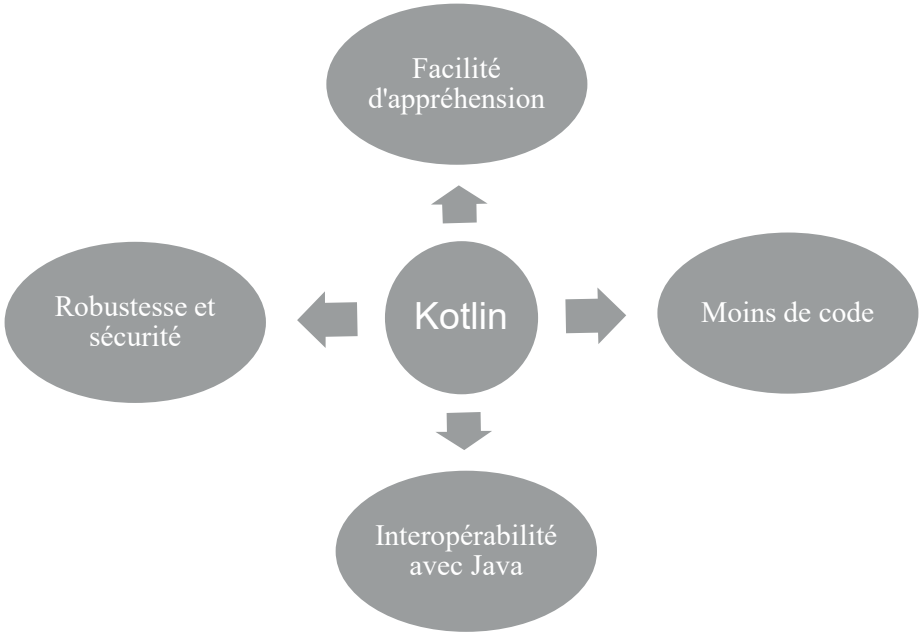
1.3 Avantages

Kotlin apporte véritablement de nouvelles fonctionnalités par rapport au langage Java qui était avant cela le seul permettant de développer des applications Android.

Sans faire une liste exhaustive des nouvelles fonctionnalités qu'apporte Kotlin, on citera :

- un gain en lignes de code (près de 40% de lignes de code en moins par rapport à Java) et une meilleure lisibilité ;
- plus de sécurité avec la détection d'erreurs dès la compilation. On aura ainsi beaucoup moins d'exceptions pendant l'exécution du code ;
- un support officiel qui s'étoffe au fil des jours. Kotlin est devenu un langage officiel dans le monde du développement Android ;
- langage supporté par l'excellent IDE Android Studio ;
- possibilité de faire cohabiter du code Java avec du code Kotlin dans un même projet. La transition peut ainsi se faire en douceur pour passer l'écriture d'une application en Java vers le langage Kotlin.

Ce langage doit donc faciliter la tâche des développeurs avec des conséquences directes pour les utilisateurs : un langage plus robuste et plus sûr doit engendrer une diminution des bugs dans l'exécution des applications.



Chapitre 2
Environnement de développement

2.1 Android Studio – présentation de l'IDE

Un IDE est un environnement de développement (IDE pour Integrated Development Environment).

Avant Android Studio, Google proposait comme environnement de développement officiel une distribution spécifique de l'environnement Eclipse, contenant notamment le SDK (Software Development Kit) d'Android.

Android Studio est annoncé le 15 mai 2013 lors du Google I/O et une version Early Access Preview est disponible le jour même. Le 8 décembre 2014, Android Studio passe de version bêta à version stable 1.0. L'environnement devient alors conseillé par Google, et Eclipse est délaissé.

Android Studio permet principalement d'éditer les fichiers Java/Kotlin et les fichiers de configuration XML d'une application Android.

Il propose entre autres des outils pour gérer le développement d'applications multilingues et permet de visualiser rapidement la mise en page des IHM (Interface Homme Machine) sur des écrans de résolutions variées. Il intègre par ailleurs un émulateur permettant de faire tourner un système Android virtuel sur un ordinateur (AVM : Android Virtual Machine).

2.2 Site officiel de Android Studio

Le site officiel de l'IDE Android Studio :

https://developer.android.com/studio

On y trouve : la page de téléchargement de l'IDE, un lien vers des nouveautés ainsi qu'un manuel de l'utilisateur.

Logo Android Studio

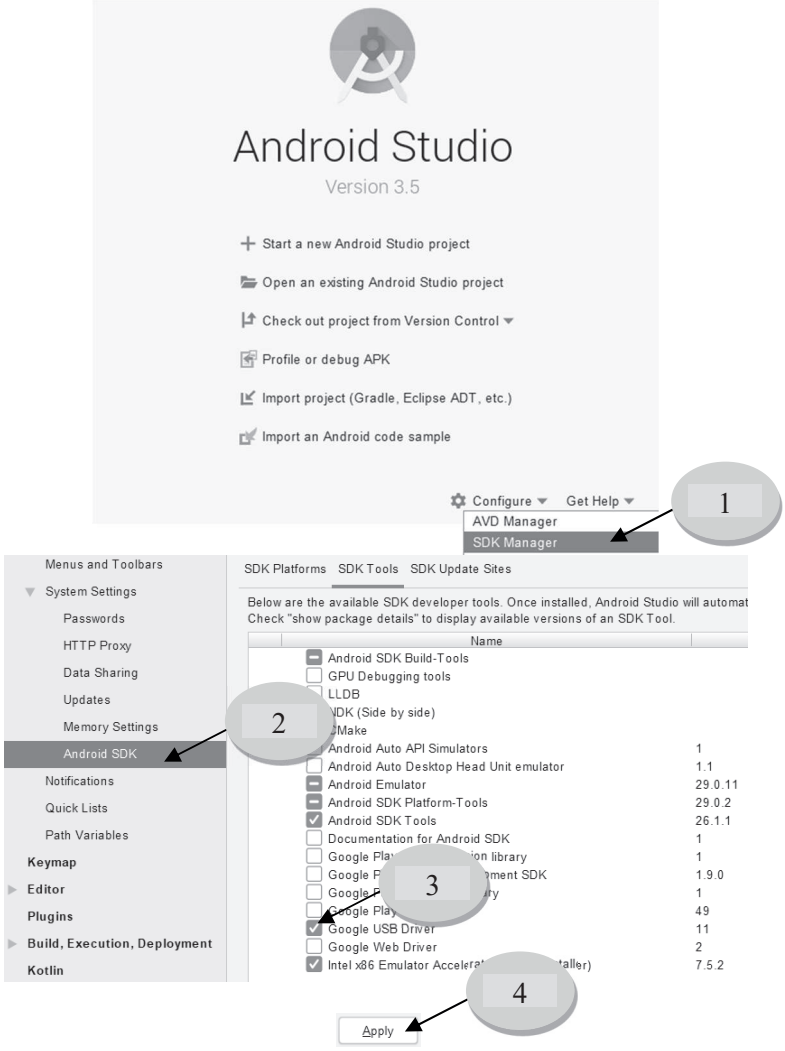
2.3 Installation d'Android Studio

Etape 1 : Installation d'Android studio

https://developer.android.com/studio

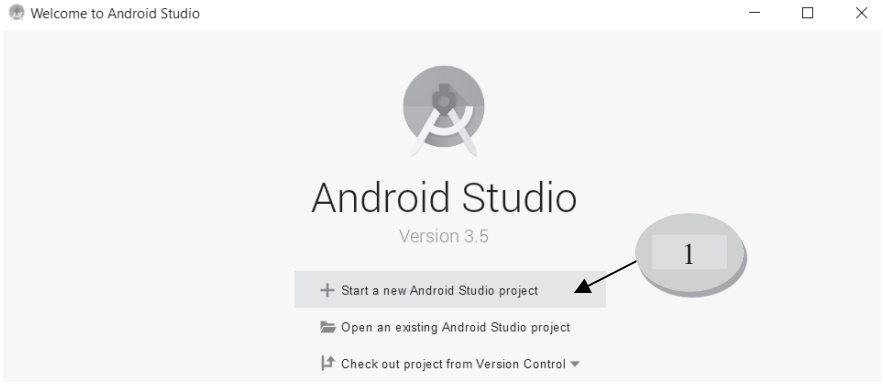
Etape 2 : Installation de google USB driver

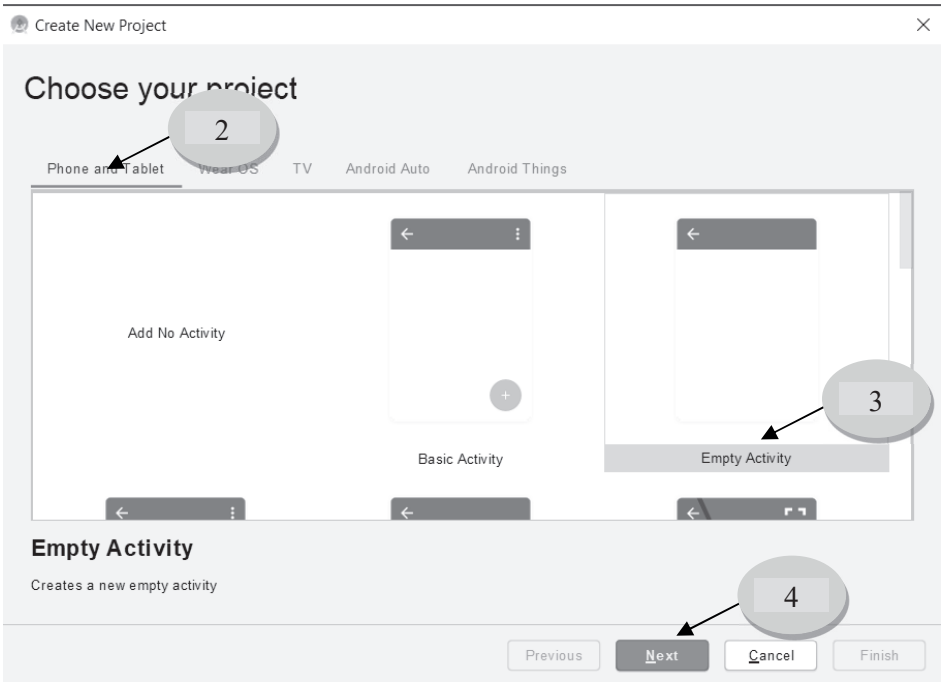
On lance Android studio puis : configure/SDK Manager, System Settings/Android SDK, onglet SDK Tools, Google USB Driver, Apply.

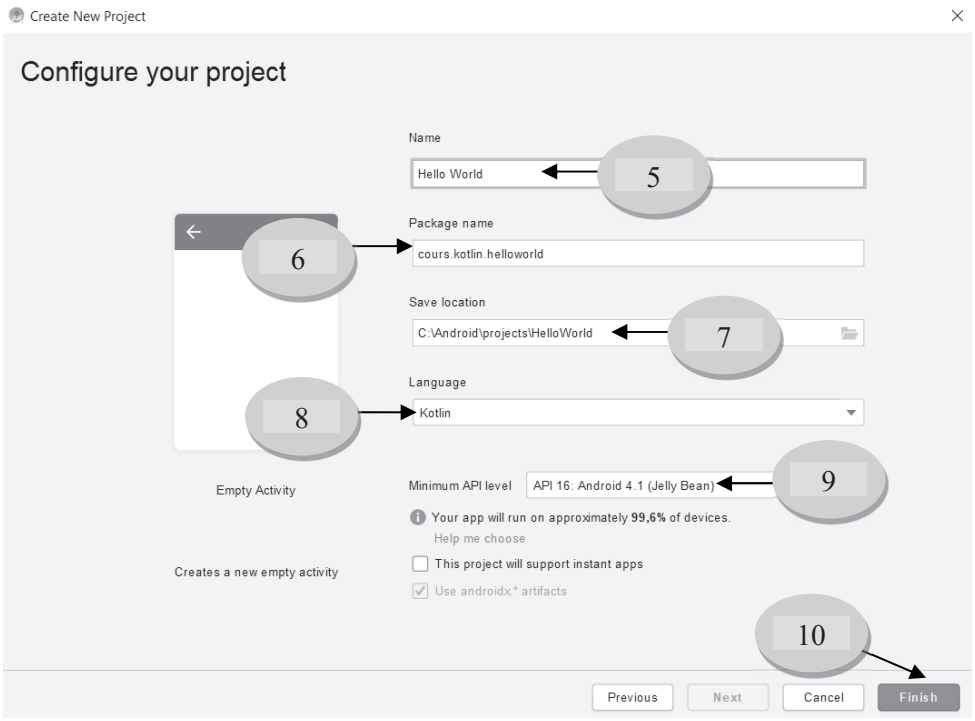


2.4 Mon premier projet

Nous allons créer un premier projet appelé Hello World.







L'étape 2 permet de choisir sur quel type d'appareil vous souhaitez faire fonctionner l'application. L'étape 3, c'est le type de l'activité principale (ici une activité vide). A l'étape 5, vous définissez le nom du projet puis le nom du package à l'étape 6. Le chemin de destination des sources du projet est indiqué à l'étape 7. Il ne faut pas oublier de choisir le langage Kotlin à l'étape 8 !

L'étape 9 permet de choisir le niveau de l'API (Application Programming Interface) minimum d'Android permettant de faire tourner votre application. Un niveau récent permet d'obtenir de nouvelles fonctionnalités mais ne tournera pas forcément sur des appareils dotés d'un API plus ancien. A vous de trouver un bon compromis entre fonctionnalités utilisées et pourcentage des appareils que vous voulez cibler.

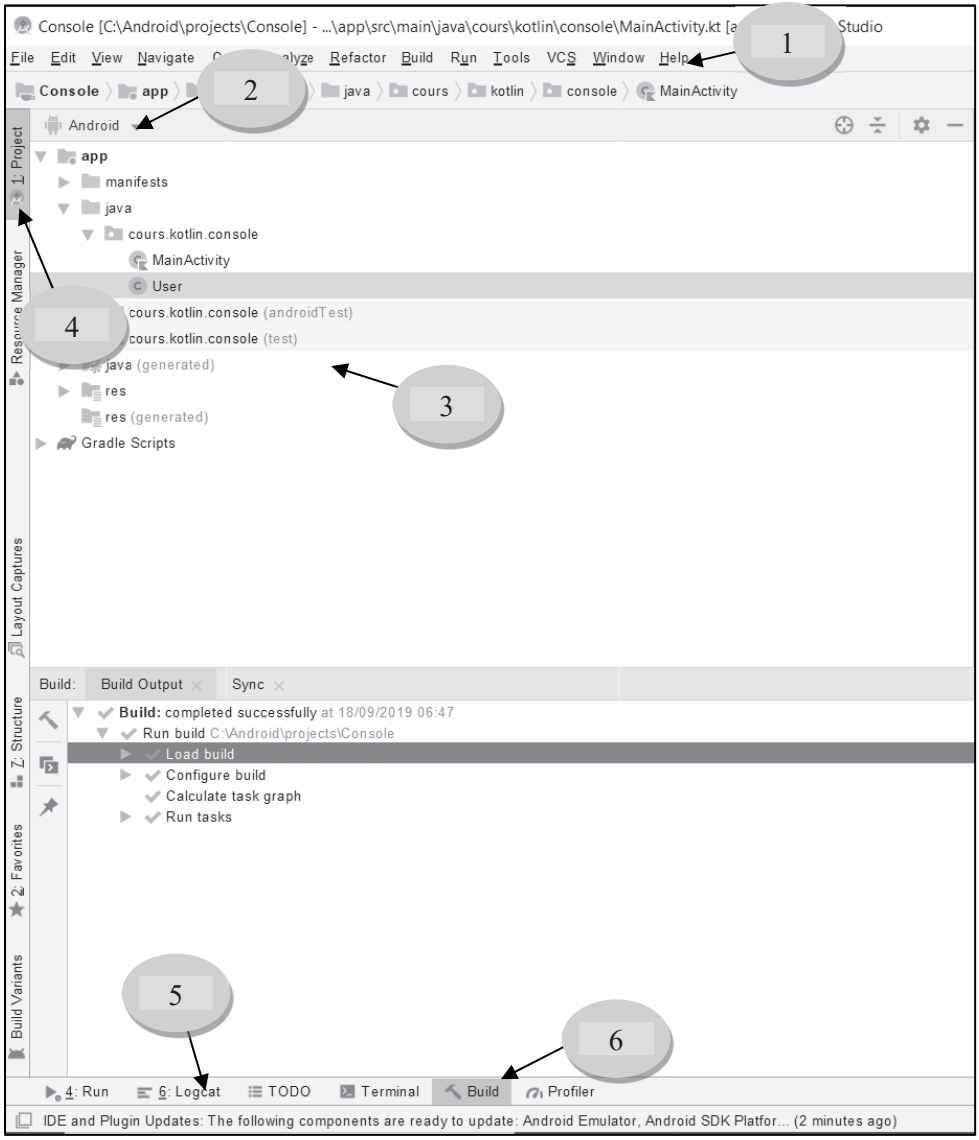
Vous cliquez enfin sur Finish et on patiente un moment : Android Studio va créer tous les fichiers ressources nécessaires au projet dans le répertoire spécifié à l'étape 7.

Une fois le processus terminé vous arrivez dans l'environnement de l'IDE et obtenez une page similaire à celle présentée au paragraphe suivant.

2.5 Fenêtre de l'IDE

Pour une meilleure lisibilité de cette présentation, la fenêtre de l'IDE Android Studio a été séparée en 2 parties :

- Partie gauche :



1) Barre des menus.

2) Choix de la vue des ressources du projet (on choisira la vue Android par défaut).

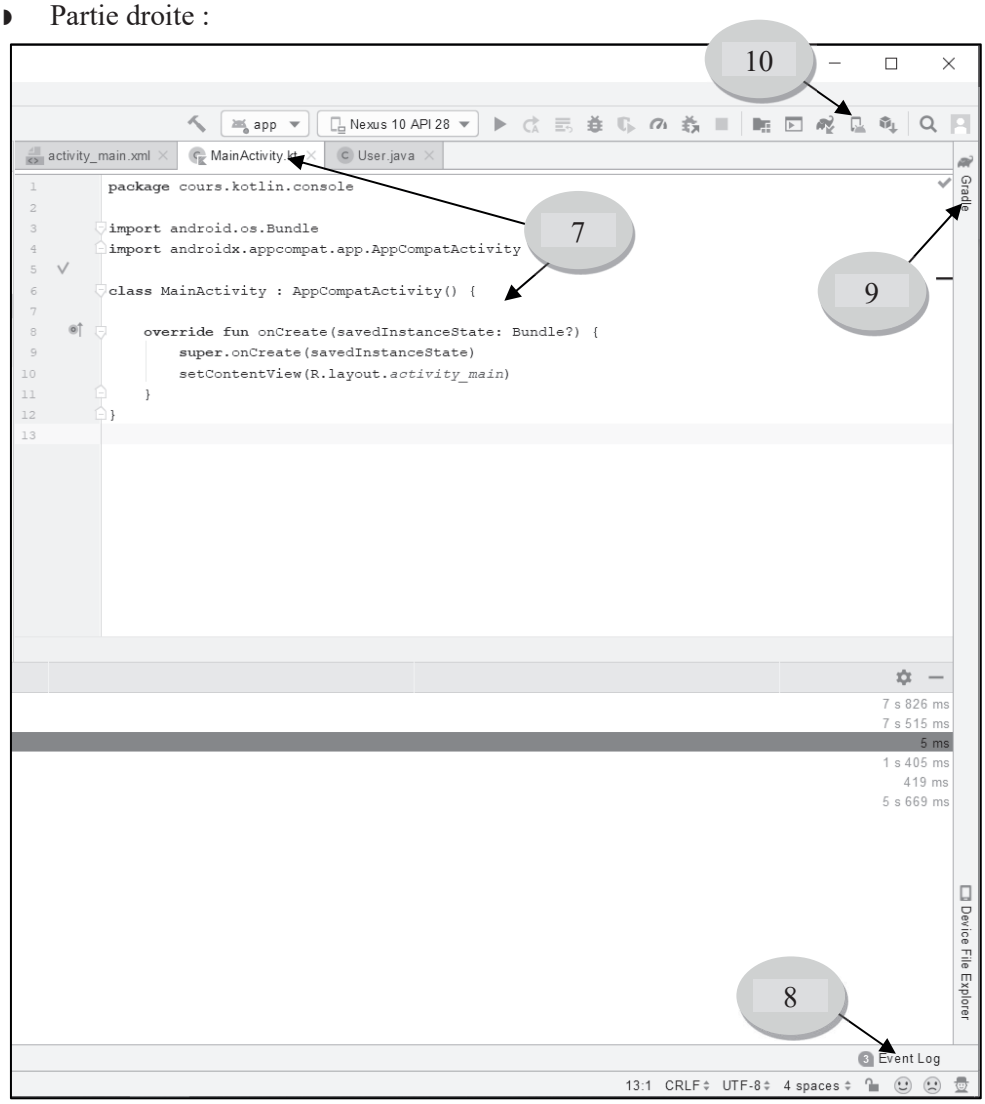
3) Fenêtre d'affichage des ressources du projet.

4) Un clic sur un onglet latéral permet de cacher (ou de visualiser) la fenêtre correspondante.

5) Le logcat utilisé pour visualiser les valeurs de sortie dans vos premiers codes.

6) Les informations sur le Build (création des ressources en début de projet, compilation, erreurs détectées dans votre code…)

- Partie droite :



7) Le fichier MainActivity.kt de notre projet. C'est ici que vous allez écrire du code kotlin.Vous observez dans cette partie un autre fichier activity_main.xml. C'est ici