Créer une application Android[[1]](#footnote-1)

Ce document explique le fonctionnement d’une application Android et montre comment développer une application pour cette plateforme.

# Qu’est-ce qu’Android?

Android est un système d’exploitation qui est sur le marché depuis 2009. Beaucoup d’eau a coulé sous le pont entre la version Cupcake et aujourd’hui. On peut dire que ce système d’exploitation est devenu mature depuis ses débuts.

Une application Android est programmée en Java et compilée dans un package de distribution en tant que APK. Le fichier APK est l’exécutable qui s’installe sur Android. Voici les différents blocs de construction d’une application Android.

Activities

Views

Intents

Requis dans toutes les applications

Services

Content Provider

Broadcast Receiver

Optionnel

Blocs de construction d’une application Android

Les blocs de base sont Activités, Vues et Intentions. Ils sont obligatoires pour toutes applications Android.

Services, Content Provider et Broadcast Receiver sont des caractéristiques avancées d’une application Android.

## Activité (Activities)

C'est la première composante Android vous rencontrerez dès que vous ouvrez une application Android. Une application Android devrait avoir au moins une activité en elle.

* En quoi consiste exactement l'activité? Chaque écran dans une application Android est une activité. Activité aura toujours une interface utilisateur (en abrégé UI). Il n'est pas possible d'avoir une activité sans une interface utilisateur.
* Toute application Android a une ou plusieurs activités. Parmi ceux-ci, une seule activité servira de point d'entrée. L'activité sera affichée lorsque l'application démarre. Cela est généralement considéré comme l'activité de lanceur ou l'activité principale aussi.
* Chaque activité a son propre cycle de vie. Vous pouvez en lire plus à ce sujet dans la [leçon Activité Concept](http://www.codelearn.org/android-tutorial/android-activity).
* Chaque application Android dispose d'un Manifest.xml où toutes les activités sont définies parmi celles-ci une activité est marquée comme activité principale. L'une des erreurs les plus communes lorsqu’un développeur débute en Android est qu’il oublie d'ajouter une nouvelle activité dans le fichier Mainfest.xml.

En tant que développeur, l'activité est un fichier de classe Java où vous écrivez la logique. Activité n'inclut pas l'interface utilisateur. Toutefois, l'une des choses que vous devez écrire dans votre logique d'activité est quelle interface utilisateur afficher.

[Tutoriel sur les activités.](http://www.codelearn.org/android-tutorial/twitter/beginner-activity-layout-example)

## Interfaces utilisateur et vues (Views)

L'interface utilisateur (UI) est ce que l'utilisateur voit sur l'écran. L'activité a la responsabilité de définir l'interface utilisateur de l'écran. L’UI comprend principalement deux types de sous-composants soit **Views** et **Layouts** ou **ViewGroups**.

La vue, comme son nom l'indique, est le bloc de construction de base comme les boutons, étiquettes, zones de texte, etc. Le layout est le conteneur pour les éléments de la vue. Le Layout sert principalement à effectuer la mise en page pour voir les éléments de la vue. Par exemple, LinearLayout permet d’indiquer si les éléments à l’intérieur sont empilés horizontalement ou verticalement tandis que RelativeLayout permet à chaque élément de la vue de se positionner par rapport à la mère ou un élément frère.

UI est défini en XML. L'élément XML supérieure est un élément de la mise en page (Layout). À l'intérieur, il peut y avoir soit des éléments de vue ou des éléments de mise en page. Un exemple de fichier XML est montré ci-dessous.

|  |
| --- |
| <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="horizontal" >  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Button 1" />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Button 2" />  <Button  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Button 3" />  </LinearLayout> |

Vous pouvez creuser un peu plus sur ce sujet dans les leçons suivantes : [Layout Concept Lesson](http://www.codelearn.org/android-tutorial/android-layout) et [View Concept Lesson](http://www.codelearn.org/android-tutorial/android-views).

[Tutoriel sur les vues.](http://www.codelearn.org/android-tutorial/twitter/beginner-activity-layout-example)

## Intents

Pour se déplacer d’une activité à une autre (ou d’un écran à un autre), sur l’interaction d’un utilisateur tel un clic sur un bouton ou une notification, les intents sont utilisés. Il est possible de passer des données et même des objets entiers avec les intents. En utilisant les intents, on peut aussi démarrer une autre application Android[[2]](#footnote-2).

Avec les activités, les vues et les intents, on peut créer une application Android de base. Beaucoup d’application sont développés en utilisant seulement ces trois concepts.

[Tutoriel sur les intents.](http://www.codelearn.org/android-tutorial/twitter/login-fire-intent-module)

## Services

Les services permettent d’exécuter les applications en arrière-plan. Par exemple, les lecteurs de musique, les notifications des différentes applications, le téléchargement de données, etc.

Les services n’ont pas d’UI. Pour montrer de l’information d’un service, les notifications sont utilisées.

Il y a deux façons de créer un service. Une méthode est d’attacher le service à une activité. Dans ce cas, le service est arrêté au même moment que l’application. L’autre façon est d’exécuter un service indépendant des applications. De cette façon, le service s’exécute toujours en arrière-plan même si l’application est arrêtée.

## Base de données SQLite dans Android

Android vient avec le nécessaire pour supporter SQLite. Les applications Android peuvent sauvegarder localement les données dans une BD SQLite. Ces données peuvent être utilisées hors ligne.

Une des raisons pour laquelle on sauvegarde des données localement est pour avoir une bonne expérience utilisateur. En effet, il est possible de restaurer les données plus rapidement. Prenons l’exemple d’une application Twitter, les tweets qui ont déjà été téléchargés sont sauvegardés dans la BD. Lorsque l’utilisateur ouvre sont applications, au lieu d’attendre que les données se téléchargent à partir du Web, il aperçoit immédiatement les anciens tweets qui ont été sauvegardés localement.

## Content Providers (Fournisseur de contenu)

Si votre application veut accéder à des données d’une tierce application, vous pouvez utiliser les fournisseurs de contenu.

Un exemple simple de l’utilisateur d’un fournisseur de contenu est l’application Contacts. Vous pouvez obtenir les contacts dans plusieurs applications tels que les SMS, le logiciel de courriel, etc.

Si vous utilisez une BD dans votre application, vous pouvez soit y accéder directement ou via un fournisseur de contenu.

## Notifications

Les notifications permettent d’afficher des messages dans la barre supérieure l’accueil Android.

* On utilise les notifications pour afficher des messages simples qui n’ont pas besoin d’une attention immédiate de la part de l’utilisateur.
* Généralement, les notifications ont un texte et une petite icône. Cependant, les dernières versions d’Android permettent d’avoir du contenu plus riche.

Les notifications sont pratiquement utilisées partout tant pour les SMS, les appels manqués, les réseaux sans fil disponibles, etc.

## Broadcast receivers

Les broadcast receivers permettent d’écouter les évènements systèmes qui arrivent sur le téléphone ou la tablette. En utilisant les broadcast receivers, on peut faire des applications tels qu’un identifieur de numéro de téléphone, un bloqueur de SMS, etc.

Voici quelques exemples d’évènement que l’on peut recevoir :

* Lorsqu’un SMS ou un appel est reçu
* Une pile faible
* L’état du réseau a changé
* Une photo a été prise par la caméra
* Le téléphone démarre

On peut aussi générer notre propre message à diffuser dans l’application.

Il est bien de noter que les broadcast receivers fonctionne même lorsque l’application n’est pas exécutée. Ceci permet d’ouvrir l’application lorsqu’un évènement survient.

1. Source du document : <http://www.codelearn.org/android-tutorial/android-introduction> [↑](#footnote-ref-1)
2. Il est aussi possible de démarrer un service. [↑](#footnote-ref-2)