

A large, semi-transparent green Android robot head and torso is positioned behind the title text.

# Introduction à la programmation Android



# Intervenant

Nathaniel Vaur Henel

Tech Lead et formateur Java

Créateur de contenu <https://nathaniel-vaur-henel.github.io/>



# Contenu du cours

21 heures de cours surtout sous forme de TP

1. Présentation générale d'Android
2. Installation de l'environnement de développement
3. Création d'une première application



# 1. Présentation générale d'Android



# Architecture

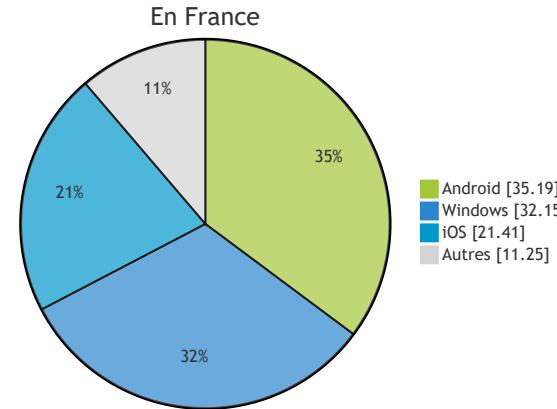
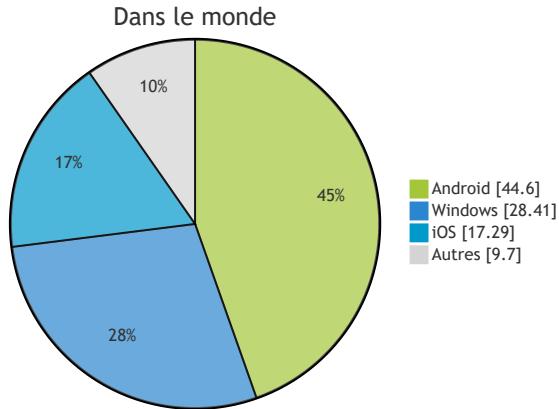
Source developer.android.com:



# Un peu d'économie

Source : Leptidigital

Les parts de marchés en 2023 des systèmes d'exploitation mobiles :



# Un peu d'histoire

- Android est né en 2004, est racheté par Google en 2005
- La version 1.5 est publiée en 2007
- De nombreuses versions depuis.
- On en est à la version 14 (octobre 2023) et l'API 34.
- La version 11 est le numéro pour le grand public, et les versions d'API sont pour les développeurs.
  - Exemples :
    - 4.1 JellyBean = API 16,
    - 6.x Marshmallow = API 23,
    - 13 = API 33

API : (Application Programming Interface) un ensemble de bibliothèques de classes pour programmer des applications.

Son numéro de version donne un indice de ses possibilités.



# Remarque sur les versions d'API

Chaque API apporte des fonctionnalités supplémentaires.

Il y a compatibilité ascendante.

Certaines fonctionnalités deviennent *dépréciées* au fil du temps, mais restent généralement disponibles.

On souhaite toujours programmer avec la dernière API (fonctions plus complètes et modernes),  
mais les utilisateurs ont souvent des smartphones plus anciens, qui n'ont pas cette API.

Or Android ne propose **aucune** mise à jour majeure.

Les smartphones restent toute leur vie avec l'API qu'ils ont à la naissance.

Les développeurs doivent donc choisir une API qui correspond à la majorité des smartphones existant sur le marché.



# Popularité des différentes versions en juin 2023

Source Wikipedia

Version	Nom de code	Date de sortie	Version API	%
4.4	KitKat	31 octobre 2013	19	0,5 %
5.0	Lollipop	31 novembre 2014	21	0, %
5.1	Lollipop	9 mars 2015	22	1,5 %
6.0	Marshmallow	8 octobre 2015	23	2,3 %
7.0	Nougat	22 août 2016	24	1,5 %
7.1	Nougat	4 octobre 2016	25	1,5 %
8.0	Oreo	21 août 2017	26	2,2 %
8.1	Oreo	6 décembre 2017	27	6,1 %
9	Pie	1er décembre 2018	28	11,9 %
10	Android 10	3 septembre 2019	29	17,8 %
11	Android 11	8 septembre 2020	30	23,1 %
12	Android 12	4 octobre 2021	31	16,3 %
13	Android 13	15 août 2022	33	15 %



# Popularité des différentes versions en juin 2023

Source Wikipedia



# Conséquences

Évolution et obsolescence voulues et très rapides.

- Suivre les modes et envies du marché, réaliser des profits
- Ce que vous allez apprendre sera rapidement dépassé (1 an) :
  - syntaxiquement (méthodes, paramètres, classes, ressources... )
  - mais pas les grands concepts (principes, organisation...) qu'on retrouverait aussi sur iOS
- autoformation permanente, mais c'est le lot des développeurs.



# Types d'applications

3 types d'applications :

- **native** : application développée en Java, C++, Kotlin, compilée et fournie avec ses données sous la forme d'une archive Jar (fichier APK). *C'est ce qu'on étudiera ici.*
- **web app** : application pour navigateur Internet, développée en HTML5, CSS3, JavaScript, dans un cadre logiciel (framework) tel que Node.js, Angular ou React.
- **hybride** : application développée dans un framework comme Ionic, Flutter, React Native... Ces frameworks font abstraction des particularités du système : la même application peut tourner à l'identique sur différentes plateformes (Android, iOS, Windows, Linux...).

Les applications **natives** sont généralement plus rapides, plus stables, plus riches en fonctionnalités, mais plus longues à développer et plus coûteuses.



# Applications natives

Une application native Android est composée de :

- **Sources Java** (ou **Kotlin**) compilés pour une machine virtuelle appelée « ART », amélioration de l'ancienne machine « Dalvik » (versions  $\leq 4.4$ ).
- Fichiers appelés **ressources** :
  - format XML : interface, textes...
  - format PNG : icônes, images...
- **AndroidManifest** : configuration de l'application
  - version minimale du smartphone,
  - demandes d'autorisations, durée de validité, etc.
  - informations sur l'application (nom, icône, thème, etc.)

Tout cet ensemble est géré à l'aide d'un IDE (environnement de développement) appelé Android Studio qui s'appuie sur un ensemble logiciel (bibliothèques, outils) appelé SDK Android.



# Kotlin

- Langage de programmation orienté objet et fonctionnel, développé par JetBrains (éditeur de l'IDE IntelliJ IDEA) et rendu open source en 2012.
- Langage officiel pour Android depuis 2017.
- Il est conçu pour être compatible avec Java, et peut être compilé en bytecode Java ou en JavaScript.
- On peut utiliser Kotlin et Java indifféremment dans un même projet.

Nous ne verrons pas Kotlin dans ce cours :

- Kotlin est un langage avancé avec une syntaxe particulièrement concise.
- Ce qui est valable pour Java l'est avec Kotlin.



# Déploiement

- Génération d'une archive signée.
- Utilisation des plateformes de téléchargement.
- Déploiement manuel en donnant l'archive `.apk` .

Si vous voulez, vous pourrez déployer votre appli sur votre smartphone grâce au mode `développeur` .



A large, semi-transparent green Android robot head and torso is positioned behind the text, partially obscuring it.

## 2. Installation de l'environnement de développement

# SDK Android

Software Development Kit

- Des librairies Java pour créer des logiciels
- Des outils de mise en boîte des logiciels
- AVD (Android Virtual Device) : Un émulateur de tablettes pour tester les applications.
- ADB (Android Debug Bridge) : Un outil de communication avec les vraies tablettes.



# Android Studio

- Un éditeur de sources et de ressources
- Un outil de compilation : gradle
- Des outils de test et de mise au point

Utilisation d'IntelliJ IDEA, l'IDE de JetBrains, possible.



# Installation

- Télécharger Android Studio sur [developer.android.com](https://developer.android.com/studio)
- Installer Android Studio
- Installer les composants nécessaires (SDK, AVD, ADB)
- Installer l'émulateur avec le device manager si besoin.

 API du device

Vous pouvez passer votre téléphone en mode développeur pour le connecter à Android Studio.

Pour cela, il faut taper 7 fois sur le numéro de version dans les paramètres.

Si cela ne fonctionne pas, cherchez sur internet.



A large, semi-transparent green Android robot head and torso is centered in the background. It has two black circular eyes, a small black antenna on its head, and a simple body. In front of it, the title text is displayed.

# 3. Crédit d'une première application

# Préparation

- Identification ou recueil des besoins
- Conception initiale et wireframing
- Choix du SDK et de l'API
- Créer l'application



# Notre projet

Une application de type Quizz.

Contiendra 3 écrans :

- Un écran de démarrage
- Un écran de jeu
- Un écran de fin de jeu



# Création du projet

Utiliser votre IDE pour créer un nouveau projet :

- Choisir un projet Android / Phone and Tablet / **No Activity**
- Name : **Quizz**
- Package name : **fr.decouverte.quizz**
- Language : **Java**
- Save location : au choix
- Minimum SDK : cf. `Help me choose`

GO !



A large, semi-transparent green Android robot head and torso is positioned behind the text.

# 3.1 Structure d'un projet Android



# Arborescence



# Arborescence

À la racine du projet :

- `build.gradle` (Projet) : fichier de configuration de Gradle pour tout le projet.
- `settings.gradle` : fichier qui indique quels modules sont inclus dans le projet.

Dossier `app` :

Contient l'application en elle-même

- `build.gradle` (Module : app) : fichier de configuration de Gradle pour l'application.

Dossier `app/src`

- Contient le code source de l'application.



# Arborescence - app/src/main

- `AndroidManifest.xml` : fichier de configuration principale de l'application. Il déclare les activités, les permissions, etc.
- `java` : contient le code source Java/Kotlin.
  - `fr.decouverte.quizz` : dossiers correspondant au package de votre application.
    - `MainActivity.java` : fichier de code de l'activité principale (et d'autres activités).
- `res` : contient les ressources de l'application.
  - `drawable` : images et graphiques utilisés dans l'application (PNG, JPEG, XML, etc.).
  - `layout` : fichiers de mise en page XML définissant l'interface utilisateur des activités et fragments.
    - `activity_main.xml` : fichier de mise en page pour l'activité principale.
  - `mipmap` : icônes de l'application (différentes tailles pour différents appareils).
  - `values` : fichiers de valeurs XML, comme `strings.xml` pour les chaînes de caractères, `colors.xml` pour les couleurs, `styles.xml` pour les styles.
  - `xml` : fichiers XML génériques utilisés pour différentes configurations (ex. `preferences.xml` pour les préférences de l'application).



```
Quizz/
├── build.gradle (Projet)
├── settings.gradle
└── app/
    ├── build.gradle (Module : app)
    └── src/
        └── main/
            ├── AndroidManifest.xml
            ├── java/
            │   └── fr/
            │       └── decouverte/
            │           └── quizz/
            │               └── MainActivity.java
        └── res/
            ├── drawable/
            │   └── ic_launcher.png
            ├── layout/
            │   └── activity_main.xml
            ├── mipmap/
            │   └── ic_launcher.png
            ├── values/
            │   ├── strings.xml
            │   ├── colors.xml
            │   └── styles.xml
            └── xml/
                └── preferences.xml
```



# Arborescence - Récapitulatif

- `build.gradle` (Projet) : configurations globales du projet.
- `settings.gradle` : modules inclus dans le projet.
- `build.gradle` (Module : app) : configurations spécifiques au module de l'application.
- `AndroidManifest.xml` : configuration de l'application (activités, permissions, etc.).
- `java` : code source Java/Kotlin de l'application.
- `res` : ressources de l'application (images, mises en page, chaînes de caractères, etc.).



A large, semi-transparent green Android robot logo serves as the background for the title. The robot is standing upright, facing slightly to the right, with its arms slightly bent at the elbows. It has a rounded head with two black dots for eyes and a small vertical line for a mouth. Its body is a dark green cylinder, and its limbs are simple green shapes.

# AndroidManifest



# AndroidManifest

Configuration et description de l'application

## 1. Déclaration des composants de l'application

- `<activity>` : Activités
- `<service>` : Services
- `<receiver>` : Récepteurs de diffusion
- `<provider>` : Fournisseurs de contenu

## 2. Permissions requises.

## 3. Configuration des composants

## 4. Informations sur l'application (Nom, icône, thème, etc.)



```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="fr.decouverte.quizz">
    <!-- Permissions requises par l'application --&gt;
    &lt;uses-permission android:name="android.permission.INTERNET" /&gt;
    &lt;uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" /&gt;
    <!-- Informations générales sur l'application --&gt;
    &lt;application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme"&gt;
        <!-- Déclaration de l'activité principale --&gt;
        &lt;activity android:name=".MainActivity"&gt;
            &lt;intent-filter&gt;
                &lt;action android:name="android.intent.action.MAIN" /&gt;
                &lt;category android:name="android.intent.category.LAUNCHER" /&gt;
            &lt;/intent-filter&gt;
        &lt;/activity&gt;
        <!-- Autres composants peuvent être déclarés ici --&gt;
    &lt;/application&gt;
&lt;/manifest&gt;</pre>
```



# AndroidManifest - Éléments clés

## 1. `<manifest>` :

- Racine du fichier, l'attribut `package` qui définit le nom de paquet unique de l'application.

## 2. `<uses-permission>` :

- Déclare les permissions pour l'application, comme l'accès à Internet ou la localisation.

## 3. `<application>` :

- Contient des attributs globaux et contient tous les composants comme les activités, services, etc.

## 4. `<activity>` :

- Déclare une activité, qui correspond à un écran de l'application.

Bien d'autres éléments peuvent être définis dans ce fichier.



# AndroidManifest - Récapitulatif

- **Permissions** : Assurez-vous de déclarer toutes les permissions nécessaires pour les fonctionnalités de votre application.
- **Activité principale** : L'activité principale doit avoir un intent filter avec l'action `MAIN` et la catégorie `LAUNCHER` pour apparaître dans le lanceur d'applications.

En résumé, le fichier `AndroidManifest.xml` est crucial pour définir les composants, les permissions et les configurations globales de votre application Android.



A large, friendly-looking green Android robot is standing upright against a light gray background. The robot has a rounded head with two small black dots for eyes and a simple black line for a mouth. It has four thick, cylindrical arms and legs made of the same green material. The word "Activity" is written in a large, white, sans-serif font across the center of the robot's body.

# Activity



# Activity

C'est un écran unique dans une application Android qui permet à l'utilisateur d'interagir avec l'application.

Elle se compose de deux parties :

- Une class Java (MonActivity.java) qui contient le code de l'Activity.
- Un fichier XML (activity\_mon.xml) qui contient la description de l'interface graphique de l'Activity.



# Activity - Cycle de vie

- `onCreate()` : appelée lors de la création de l'Activity. C'est ici que vous devez initialiser l'interface utilisateur.
- `onStart()` : appelée lorsque l'Activity devient visible à l'utilisateur.
- `onResume()` : appelée lorsque l'Activity commence à interagir avec l'utilisateur. Malgré son nom, elle est appelée au démarrage, même s'il n'y a rien à reprendre.
- `onPause()` : appelée lorsque l'Activity est partiellement cachée.
- `onStop()` : appelée lorsque l'Activity n'est plus visible.
- `onDestroy()` : dernière méthode appelée avant que l'Activity ne soit détruite.



# Activity - État Cycle de vie

Source : [developer.android.com](https://developer.android.com)



# Activity - Classe Java

Voici un exemple simple d'une activité en Java :

```
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;

// AppCompatActivity est une activité de base compatible avec les anciennes versions d'Android
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) { // appelée lorsque l'activité est créée pour la première fois.
        super.onCreate(savedInstanceState); // Appel du onCreate de la classe parente pour effectuer la configuration
        setContentView(R.layout.activity_main); // Définition de la mise en page de l'activité à partir du fichier XML
    }
}
```



# Activity - Fichier xml

La mise en page (layout) d'une activité est définie dans un fichier XML dans le dossier `res/layout`.

```
<!-- Déclaration de la mise en page. LinearLayout est un conteneur qui organise ses enfants en ligne ou en colonne. -->
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" "La largeur de LinearLayout s'ajuste à l'écran."
    android:layout_height="match_parent" "La hauteur de LinearLayout s'ajuste à l'écran."
    android:orientation="vertical"> "Les éléments enfants seront disposés verticalement."
<!-- Déclaration d'un TextView qui affiche un texte. -->
<TextView
    android:layout_width="wrap_content" "La largeur du TextView s'ajuste à son contenu."
    android:layout_height="wrap_content" "La hauteur du TextView s'ajuste à son contenu."
    android:text="Hello, World!" "Le texte affiché dans le TextView."
    android:textSize="18sp" "La taille du texte, en unité scalable pixels (sp), adaptée à la taille de l'écran."
    android:padding="16dp" "Ajoute un espace autour du texte à l'intérieur du TextView."
    android:layout_gravity="center_horizontal"/> "Centre horizontalement le TextView dans le LinearLayout."
<!-- Déclaration d'un Button que l'utilisateur peut cliquer.-->
<Button
    android:layout_width="wrap_content" "La largeur du Button s'ajuste à son contenu."
    android:layout_height="wrap_content" "La hauteur du Button s'ajuste à son contenu."
    android:text="Click Me" "Le texte affiché sur le bouton."
    android:onClick="handleButtonClick" "Méthode appelée lorsque le bouton est cliqué."
    android:layout_gravity="center_horizontal"/> "Centre horizontalement le Button dans le LinearLayout."
</LinearLayout>
```



# Activity - Déclaration dans manifest

Toutes les activités doivent être déclarées dans le fichier `AndroidManifest.xml` :

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="fr.decouverte.quizz" > <!-- Nom unique du package de l'application -->

    <application>
        <!-- Déclaration de l'activity nommée MainActivity -->
        <!-- exported="true" permet son interaction avec d'application, comme l'OS -->
        <activity android:name=".MainActivity"
                  android:exported="true"
            >
            <intent-filter>
                <!-- Indique que cette activity est le point d'entrée principal de l'application -->
                <action android:name="android.intent.action.MAIN" />
                <!-- Fait apparaître cette activity dans le lanceur d'applications -->
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```



A large, friendly-looking green Android robot is positioned behind the title text. It has a rounded head with two small black dots for eyes, a wide, open mouth, and four thick, rounded arms and legs. The robot is facing towards the left.

# Ressources



# Ressources

Des fichiers xml qui contiennent des données accessibles partout, dans les fichiers Java et xml.

Le tag racine est toujours `<resources>`.

- `strings.xml` : Définit des chaînes de caractères.

```
<resources>
    <string name="app_name">MonApplication</string>
    <string name="hello_world">Bonjour, le monde!</string>
</resources>
```

- `colors.xml` : Définit des couleurs.

```
<resources>
    <color name="primaryColor">#008577</color>
    <color name="primaryDarkColor">#00574B</color>
    <color name="accentColor">#D81B60</color>
</resources>
```



- `themes.xml` : Définit des styles et thèmes.

```
<resources>
    <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
        <item name="colorPrimary">@color/primaryColor</item>
        <item name="colorPrimaryDark">@color/primaryDarkColor</item>
        <item name="colorAccent">@color/accentColor</item>
    </style>
</resources>
```

- `dimens.xml` Définit des dimensions (marges, tailles de police, hauteurs, etc.).

```
<resources>
    <dimen name="activity_horizontal_margin">16dp</dimen>
    <dimen name="activity_vertical_margin">16dp</dimen>
</resources>
```



- `attrs.xml` : Définit des attributs personnalisés.

```
<resources>
    <declare-styleable name="CustomView">
        <attr name="customAttribute" format="integer"/>
    </declare-styleable>
</resources>
```

- `arrays.xml` : Définit des tableaux de valeurs, chaînes de caractères ou entiers.

```
<resources>
    <string-array name="planets_array">
        <item>Mercury</item>
        <item>Venus</item>
        <item>Earth</item>
        <item>Mars</item>
    </string-array>
</resources>
```



- `integers.xml` : Définit des valeurs de nombres entiers.

```
<resources>
    <integer name="max_attempts">5</integer>
</resources>
```

- `bools.xml` : Définit des valeurs booléennes.

```
<resources>
    <bool name="is_feature_enabled">true</bool>
</resources>
```



- `plurals.xml` : Définit les ressources de pluriels, permettant de spécifier différentes chaînes selon la quantité.

```
<resources>
    <plurals name="numberOfSongsAvailable">
        <item quantity="zero">No song available</item>
        <item quantity="one">1 song available</item>
        <item quantity="other">%d songs available</item>
    </plurals>
</resources>
```



# Activity - Récapitulatif

- **Représentation d'un écran** : Une Activity représente un écran avec lequel l'utilisateur peut interagir.
- **Cycle de vie** : Comprendre le cycle de vie est crucial pour gérer les ressources efficacement.
- **Interface utilisateur** : La mise en page est définie en XML et liée dans `onCreate()`.
- **Déclaration** : Chaque activité doit être déclarée dans `AndroidManifest.xml`.

En résumé, les activités sont le cœur de l'interaction utilisateur dans une application Android, gérant les écrans et leur cycle de vie.



A large, semi-transparent green Android robot head and torso is positioned behind the text.

## 3.2 Codons notre première Activity

# > Créons notre première Activity

Utilisons l'IDE pour créer une nouvelle activity vide : New -> Activity -> Empty Views Activity .

Cochons la case `Launcher Activity` qui ajoutera les balises `<intent-filter>` dans le manifest :

```
<activity
    android:name=".MainActivity"
    android:exported="true"
    >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Cela lui permettra d'être lancée au démarrage de l'application.



## > Testons grâce à l'émulation

L'émulation permet de lancer l'application Android sur l'ordinateur.

Le device manager permet de créer un émulateur de plusieurs types de périphériques et même de créer les siens.

Il faut choisir un device avec une API compatible avec celle de l'application.

Il est aussi possible de connecter un smartphone en mode développeur en USB ou en Wi-Fi.

Attention à la version de l'API du smartphone.



## > Regardons le code généré

Le code Java et XML ressemble à ce qu'on a déjà vu précédemment



# Une dernière chose

Pouvez-vous de remplir ce formulaire afin de m'aider à améliorer ce cours ?

<https://forms.gle/ESbxxDJUFpxSZEP17>



Merci !