IUT de Paris - Rives de Seine
Université Paris Cité

R4.Real.11 – Développement pour applications mobiles

– TP 1 –

Vues d'authentification

Pr. Chaouche A.-C.
ac.chaouche@gmail.com

Résumé

Prérequis

- Programmation Java
- Cycle de développement d'Android

Objectifs

- Manipuler l'environnement Android Studio
- Créer un nouveau projet Android
- Comprendre la notion d'activité
- Gérer les ressources graphiques
- Exécuter une application Android

Outils nécessaires pour ce cours

Android Studio. Otter 2 Feature Drop v2025.2.2 

JDK. v1.8 

Android SDK. Baklava API 36 

Android Emulator. Google Pixel 9 

Outil de développement

Android Studio. Otter 2 Feature Drop v2025.2.2 (Décembre 2025)

- Adapté à la programmation Android
- Gourmand en ressource
- Complétion de code avancée
- Intégration de l'assistant Gemini

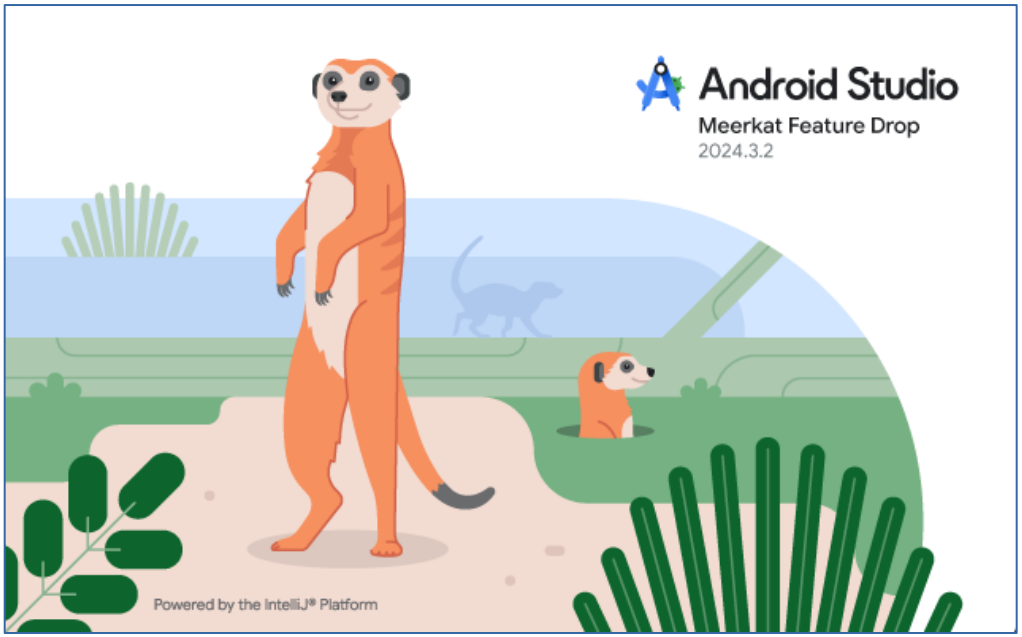
Besoins en matériel

Besoin	Minimum	Recommandé
OS	64-bit - Microsoft Windows 8 - MacOS 10.14 - Linux avec Gnome, KDE, ou Unity DE	Dernière version 64-bit
RAM	8 Go	16 Go ou plus
Processeur	x86_64 CPU architecture 2ème génération Intel Core	Dernier processeur Intel Core
Espace disque	8 Go	16 Go ou plus (SSD)
Résolution d'écran	1280 x 800	1920 x 1080

Installer Android Studio

Installer Android Studio. Otter 2 Feature Drop v2025.2.2

- **Lien :** https://developer.android.com/studio/
- **Chemin recommandé :** D:\android\android-studio



SDK Manager (1/2)

SDK Manager est un outil CLI et GUI

- Fourni dans le SDK d'Android
- Permettant de manipuler les paquets du SDK
- Afficher, installer, mettre à jour, désinstaller

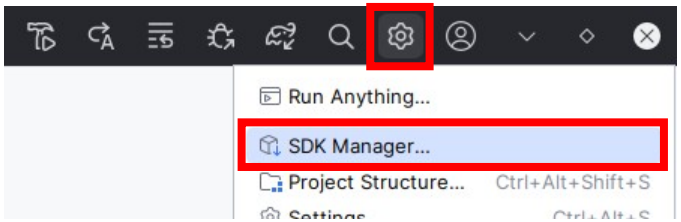
Principaux paquets :

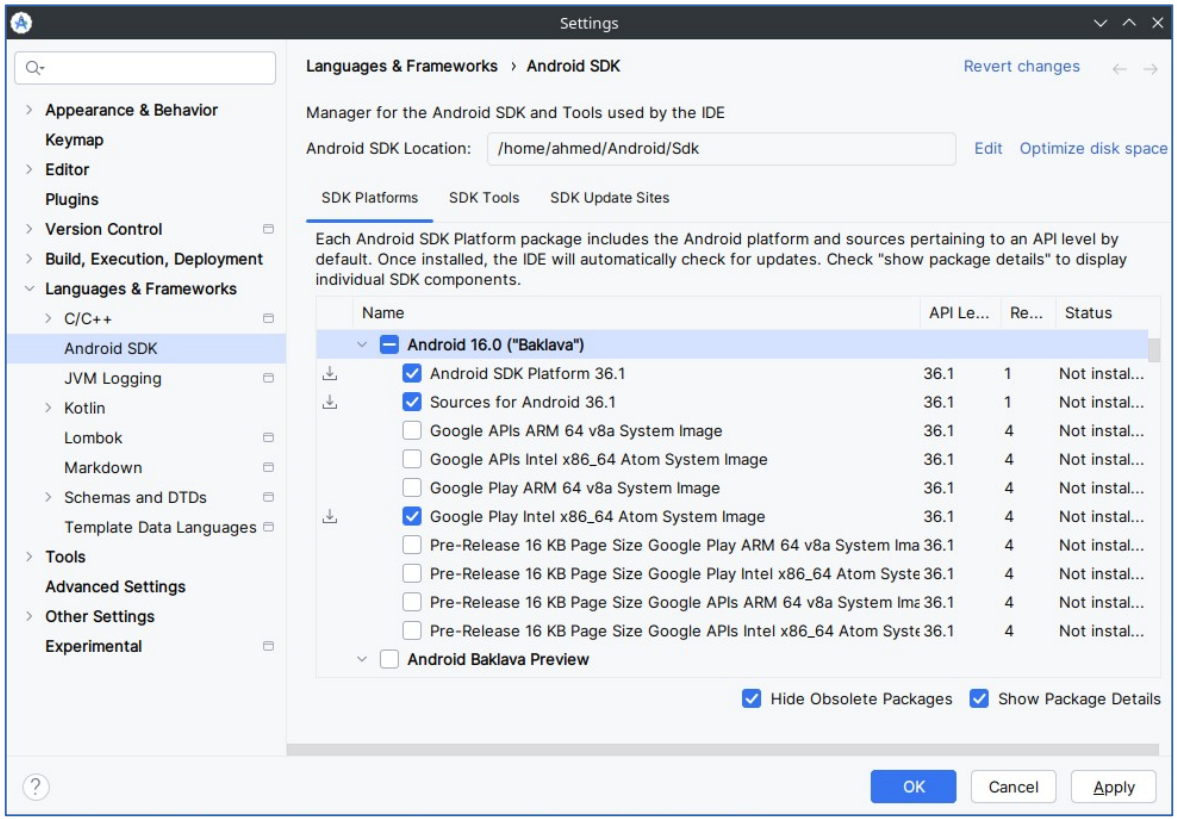
- SDK Tools (Exécution)
- SDK Build-tools (Construction)
- SDK Platform-tools (ADB)
- SDK Platforms (Versions d'Android)
- System Image (Emulateurs)
- Samples (Exemples de code)

SDK Manager (2/2)

Sous Android Studio :

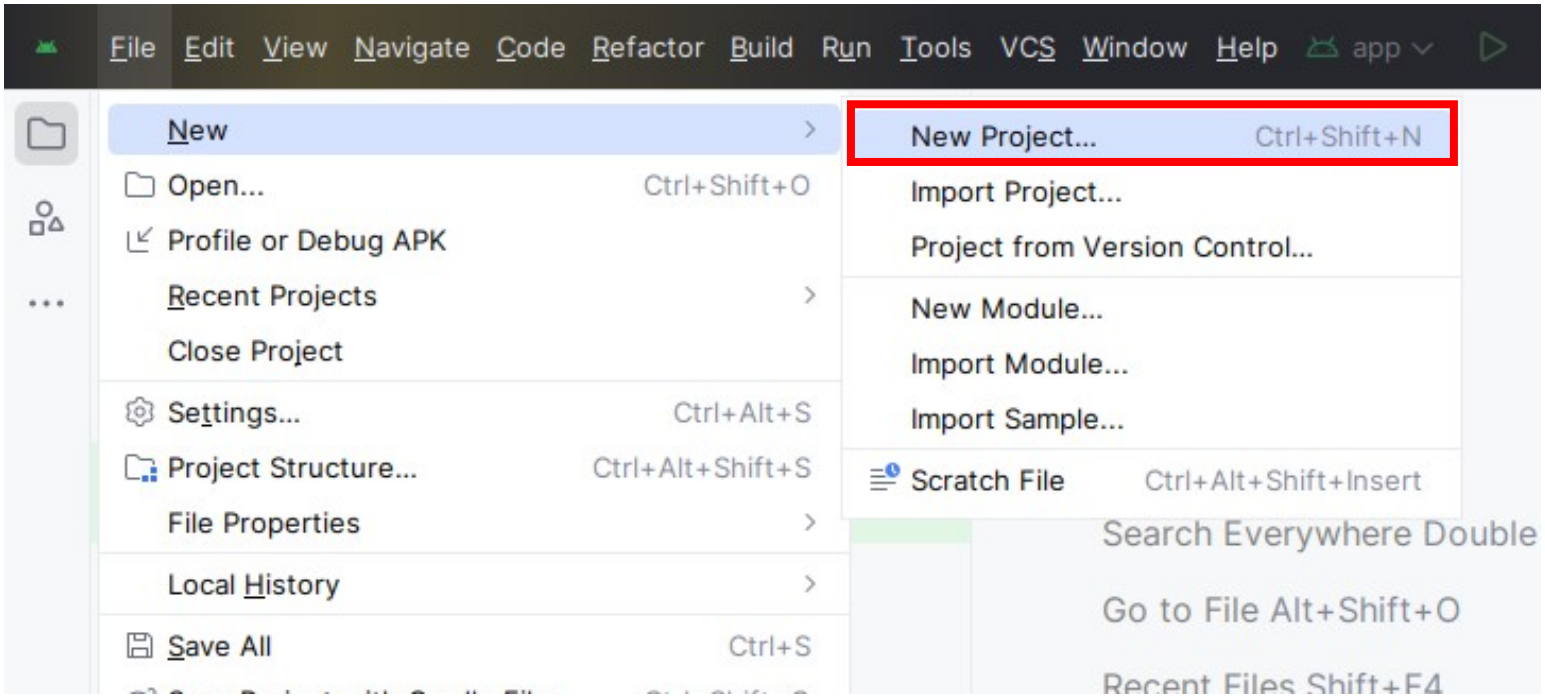
- **Tools > SDK Manager**





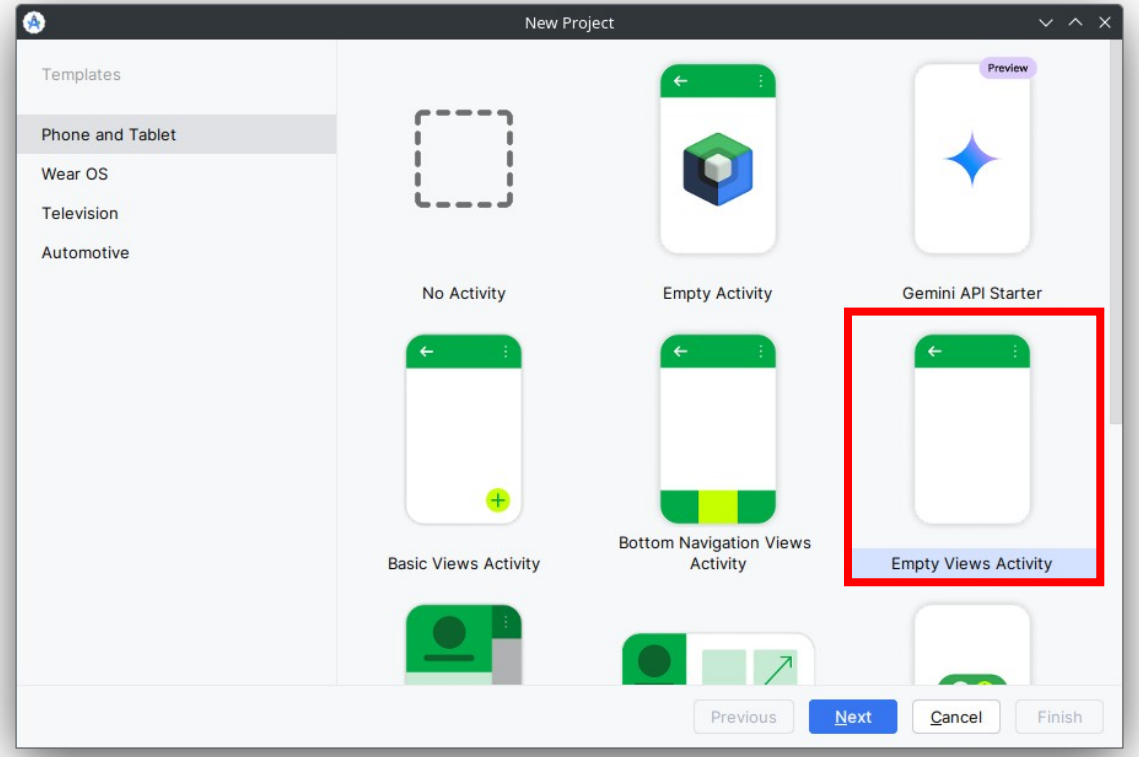
Création d'un nouveau projet (1/3)

- Procédure : **File** > **New** > **New Project...**



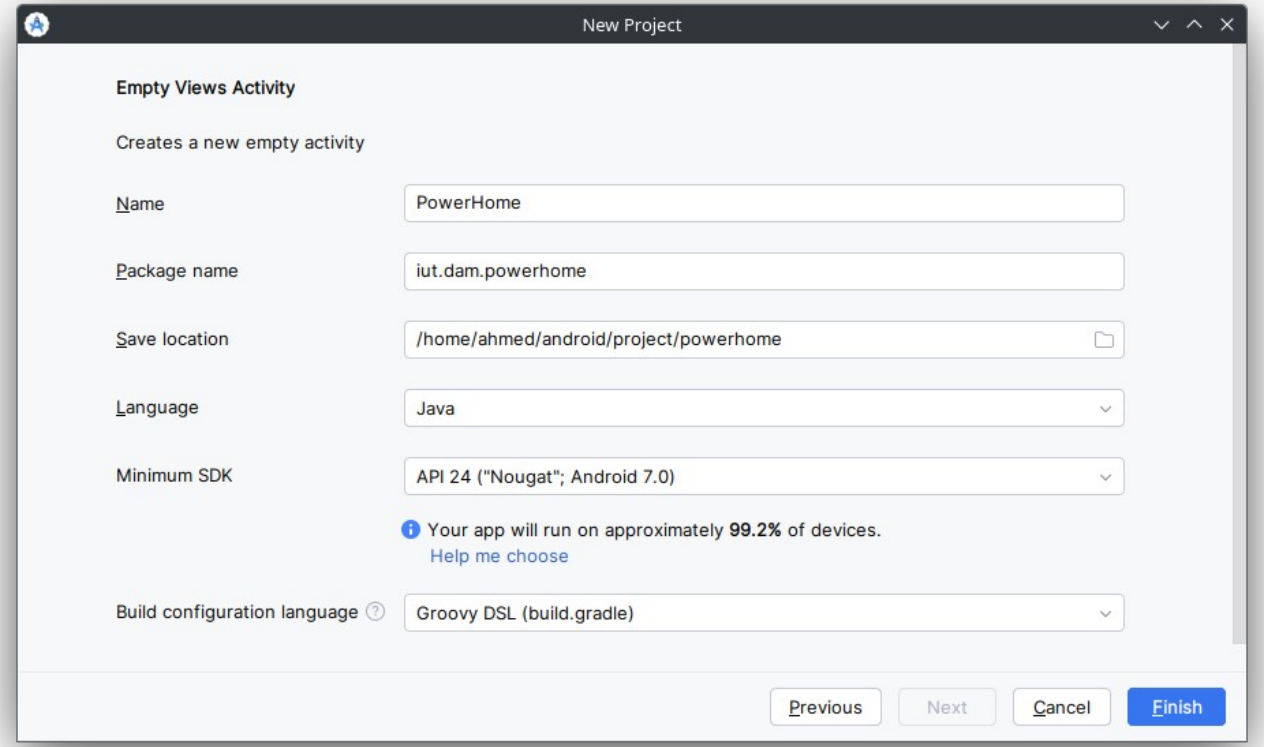
Création d'un nouveau projet (2/3)

- Type d'appareil : **Phone and Tablet**
- Template : **Empty Views Activity**

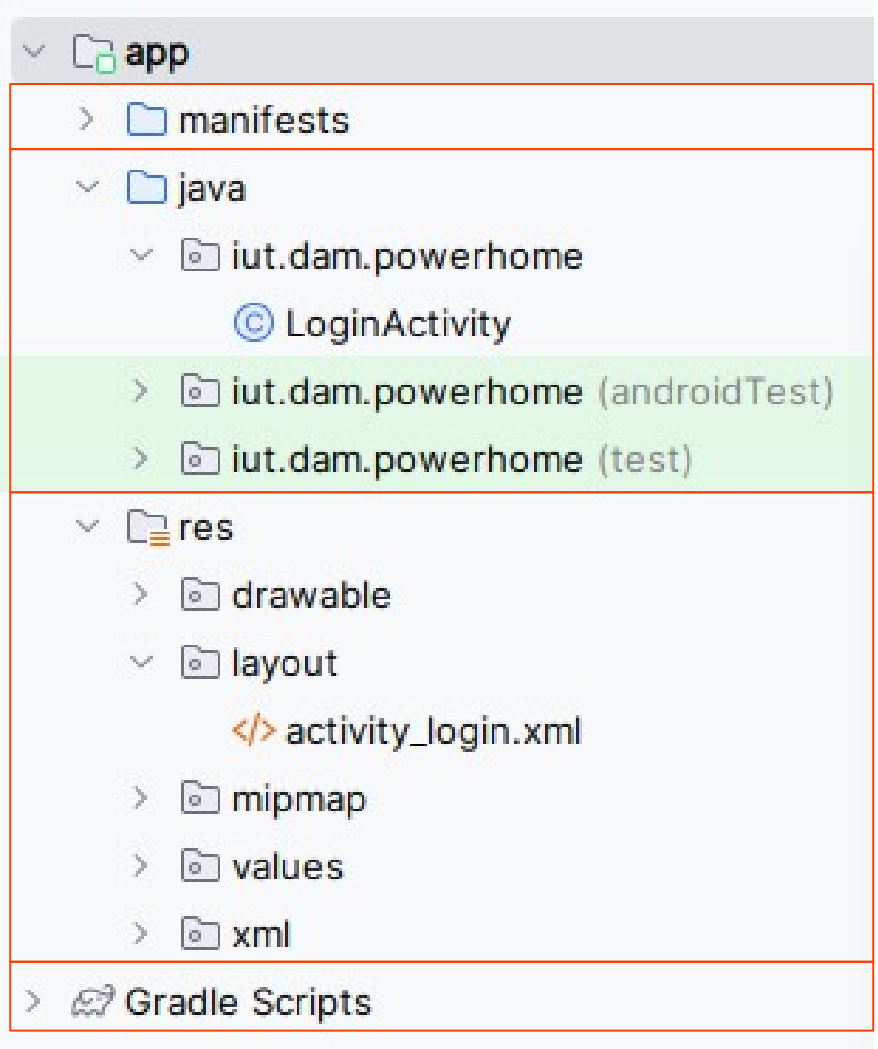


Création d'un nouveau projet (3/3)

- Application Name : **PowerHome**
- Package : **`iut.dam.powerhome`**
- Save location : **D:\android\projects**
- Language : **Java**
- Minimum SDK : **API 24** (99.2 % of devices)
- Build langage : **Groovy DSL**



Structure d'un projet



Manifest

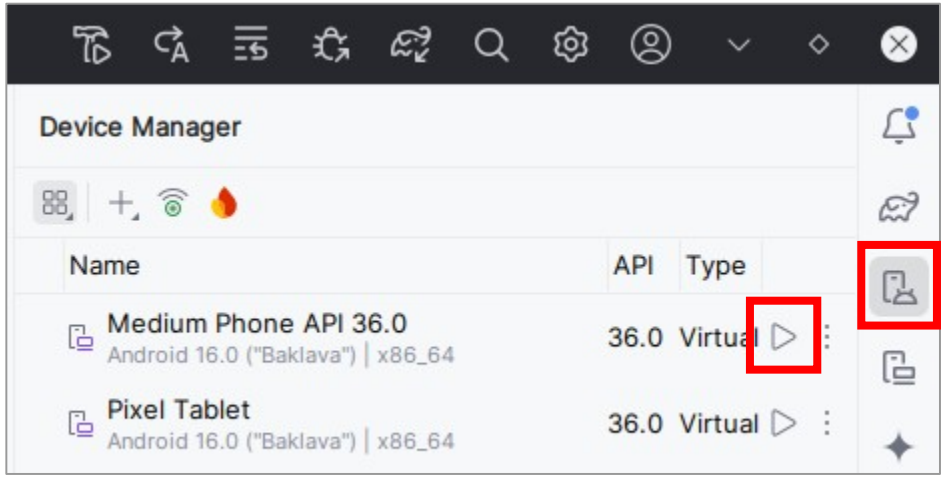
Fichiers java

Ressources organisées (texte, ...)

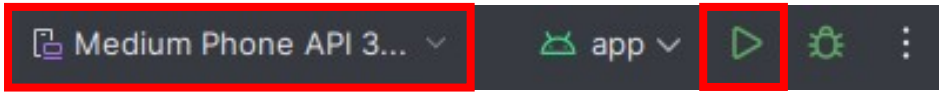
Gradle

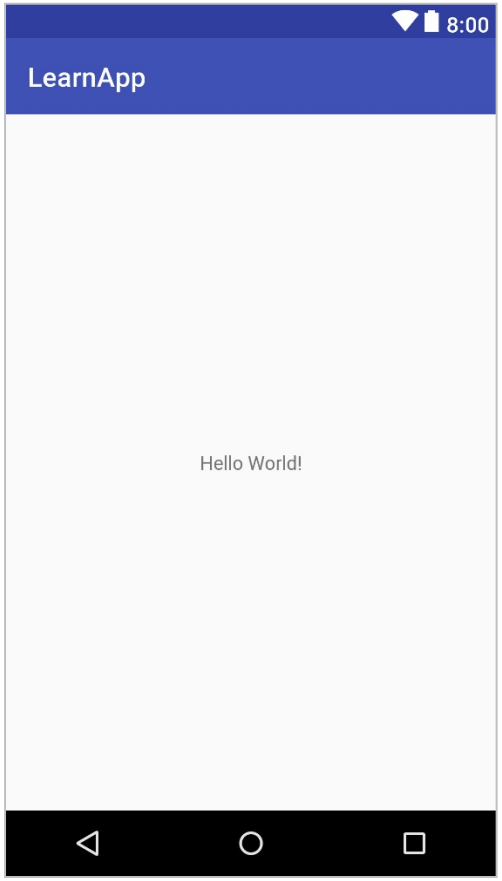
Exécution de l'application

1. Lancer l'emulateur
(ou brancher un appareil physique)



2. Sélectionner l'appareil et lancer l'exécution





"AndroidManifest.xml" (1/2)

Décrit les composants de l'application

- Définit le point d'entrée potentiel de l'application (activity main)

Quatre types de composants

- Activités (activities)
- Services (services)
- Fournisseurs de contenu (content providers)
- Récepteurs de diffusion (broadcast receivers)

Décrit les permissions requises par l'application

- Accès a l'Internet, lecture-écriture dans la liste de contacts,
- Géo-localisation, . . .

Décrit les besoins matériels et logiciels de l'application

- Appareil photo, bluetooth, ecran multi-touch, . . .

"AndroidManifest.xml" (2/2)

/manifests/AndroidManifest.xml

```
<manifest>
  <application>
    <activity android:name=".LoginActivity"
              android:label="@string/app_name">
      <intentfilter>
         <action android:name="android.intent.action.MAIN"/>
         <category android:name="android.intent.category.LAUNCHER"/>
      </intentfilter>
    </activity>
  </application>
</manifest>
```

Fichier XML obligatoire, à la racine du projet

- **<manifest>** et **<application>** sont obligatoires
- **category.LAUNCHER** : point d'entrée de l'application

Activité (1/4)

Une activité = un écran graphique

- définie dans **`./java/`**
- souvent c'est un cas d'utilisation UML
- contrôle les éléments définis par du code Java

Hérite de

- **`android.app.Activity`** ou
- **`androidx.appcompat.app.AppCompatActivity`**

> **`AppCompatActivity`** est une extension de **`Activity`** qui fournit une **compatibilité descendante** et des fonctionnalités UI modernes sur un large éventail de versions Android.

Interface graphique (IHM)

- associé à une vue (**`./res/layout/`**)
- certains éléments (texte, dimension, couleur) sont définis dans **`./res/values`**

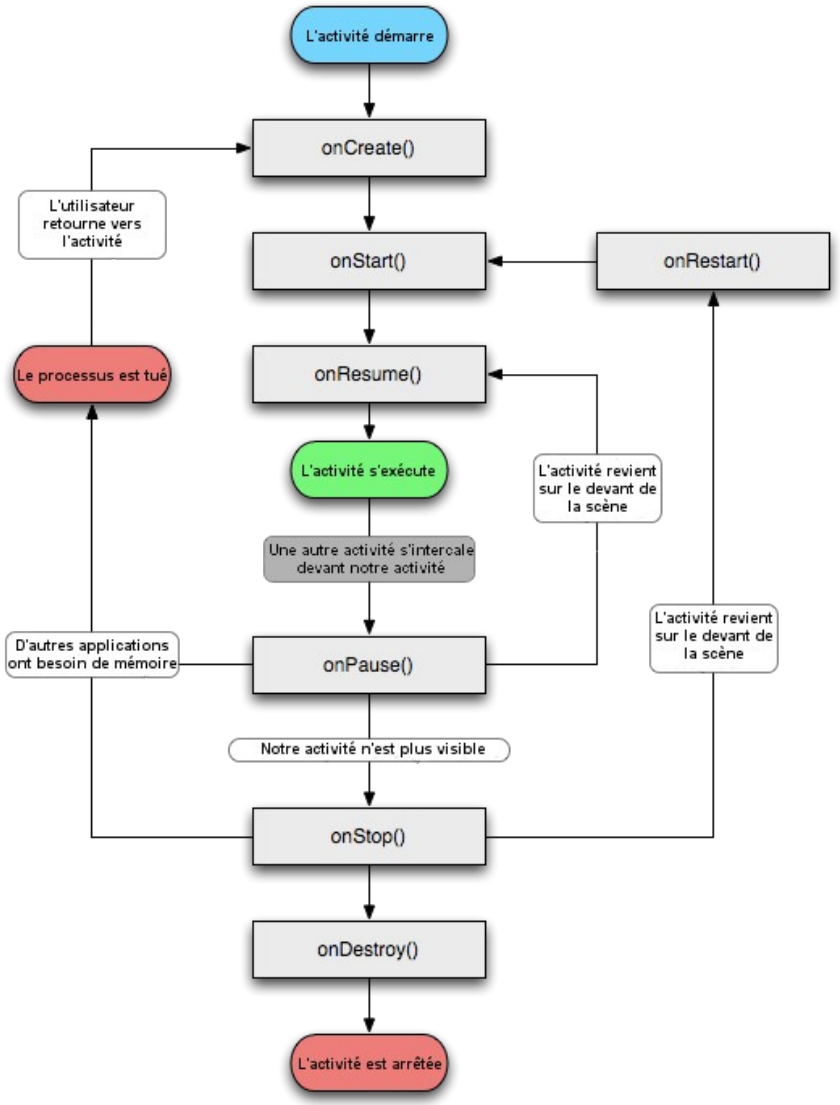
Activité (2/4)

Le **changement d'état d'une activité** provoque le déclenchement de la **méthode callback** correspondante.

Méthodes callback :

```
void onCreate(...)
void onStart()
void onRestart()
void onResume()
void onPause()
void onStop()
void onDestroy()
```

(Voir Cours 2)



Activité (3/4) – Exemple

/java/LoginActivity.java

```
import androidx.appcompat.app.AppCompatActivity;

public class LoginActivity extends AppCompatActivity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        ...
    }
    @Override
    protected void onDestroy() {
        super.onDestroy();
    }
}
```

Pour chaque méthode callback, appeler la méthode sur **`super`**

- **Exemple :** dans **`onCreate()`**, appel à **`super.onCreate()`**

Le **`Bundle`** mémorise l'état de l'activité lorsqu'elle passe en arrière-plan

Activité (4/4) – Association d'un layout

/java/LoginActivity.java

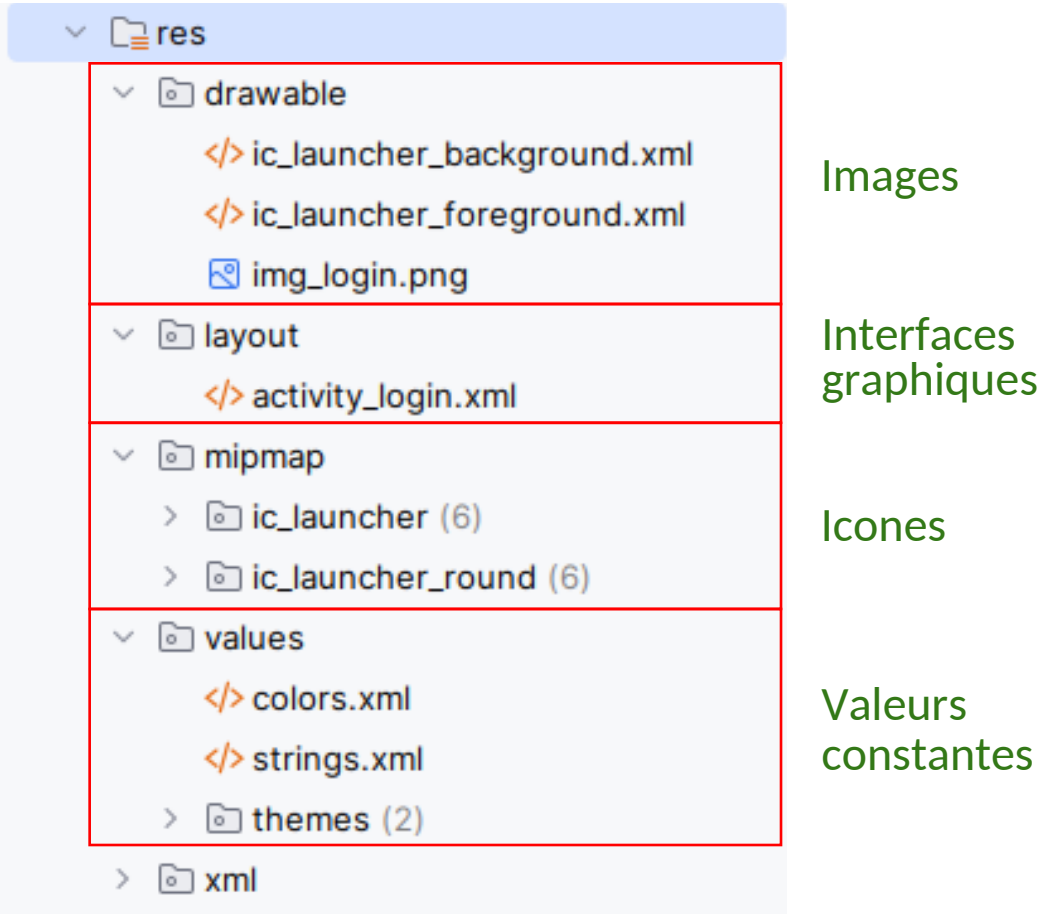
```
public class LoginActivity extends AppCompatActivity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        ...
    }
    ...
}
```

- **setContentView(int layout)** associe à l'activité un Layout référencé par **layout**

Ressources "./res/"

Types de ressources :

- **Layouts et menus**
- **Icones et images**
- **Valeurs (values) :**
  - strings, attrs,
  - styles, colors,
  - dimens,
  - …



Ressources : Layouts

Description

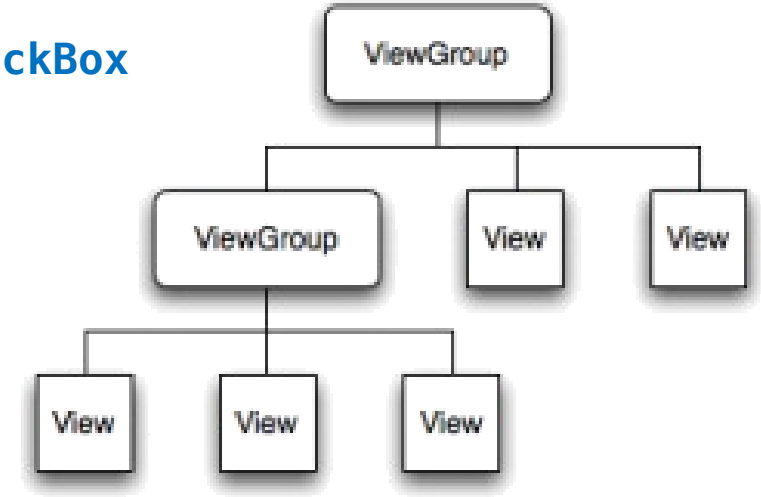
- Déclarative (Fichiers XML dans le répertoire **`./res/layout/`**)
- Ou directement dans le code java

Widget

- Composant élémentaire
- Instance de la classe **`View`**
- Sensible aux événements
- Exemples : **`Button`**, **`EditText`**, **`TextView`**, **`CheckBox`**

Layout

- Conteneur de vues
- Instance de la classe **`ViewGroup`**
- Exemples : **`LinearLayout`**, **`TableLayout`**, **`RelativeLayout`**, **`FrameLayout`**, **`ScrollView`**



Ressources : Layouts

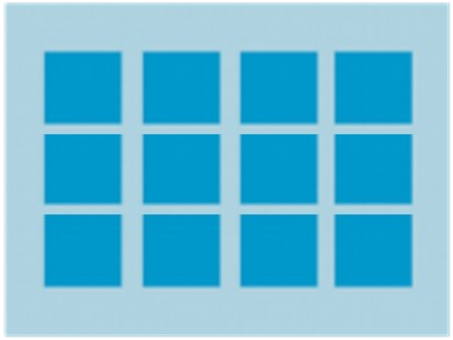
Types de **`ViewGroup`** (1/2)

`LinearLayout`

`RelativeLayout`

`WebView`

`GridLayout`



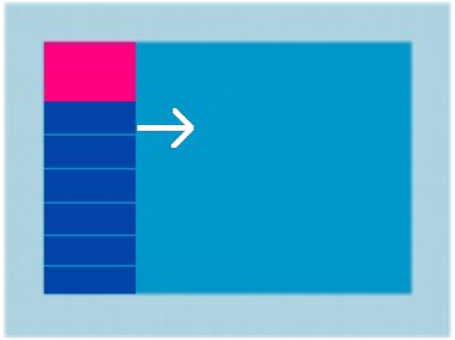
`FrameLayout`

`AbsoluteLayout`

Ressources : Layouts

Types de **ViewGroup** (2/2)

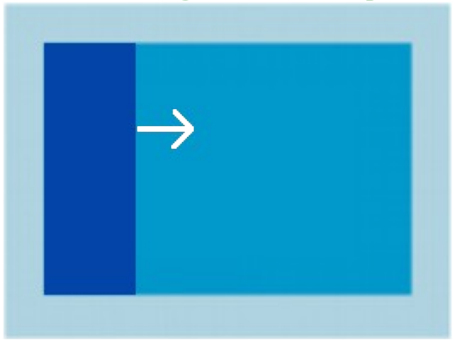
DrawerLayout



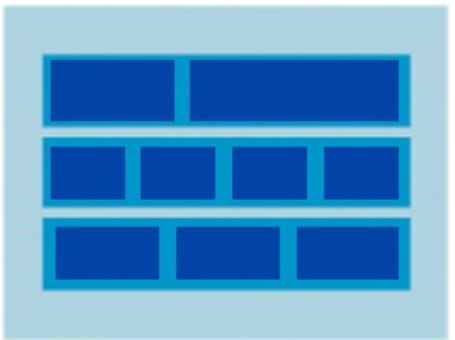
CoordinatorLayout

SwipeRefreshLayout

SlidingPaneLayou



TableLayout



TabLayout-ViewPager

Ressources : Layouts

Exemple d'un layout

/res/layout/activity_login.xml (Text)

```
<?xml version="1.0" encoding="utf8"?>
<LinearLayout
  xmlns:android="http://schemas.android.com/...
  android:orientation="vertical"
  android:layout_width="match_parent"
  android:layout_height="match_parent">
  <TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Salut IUT de Paris" />
  <TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Let's go" />
</LinearLayout>
```

Ressources : Layouts

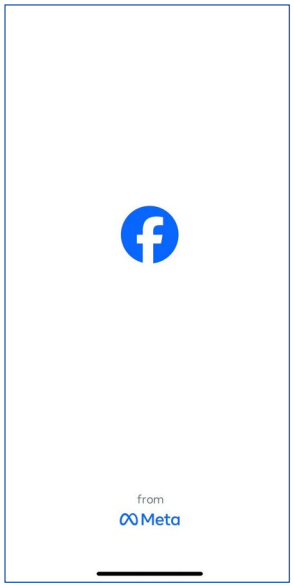
Attributs de **`LinearLayout`**

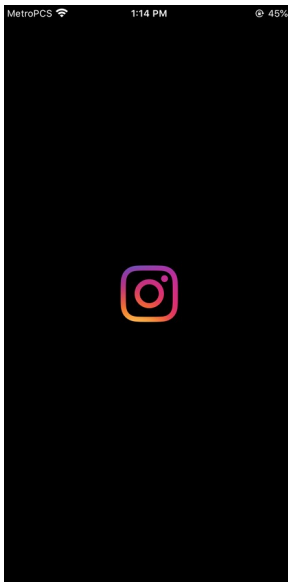
- **`orientation`** : vertical | horizontal
- **`layout_width`** et **`layout_height`** :
  - **Valeur constante :** en dp ou px
  - **`wrap_content`** **:** place minimum
  - **`match_parent`** **:** taille du parent
- **`gravity`** : top, bottom, left, right, center, …
- **`padding`** et **`margin`** : en dp ou px

TP1a : Vue de démarrage

Créer une activité `SplashActivity`

- **Un `TextView` pour afficher le nom de l'application**
- **Un logo de l'application**
- **Une palette de couleurs**





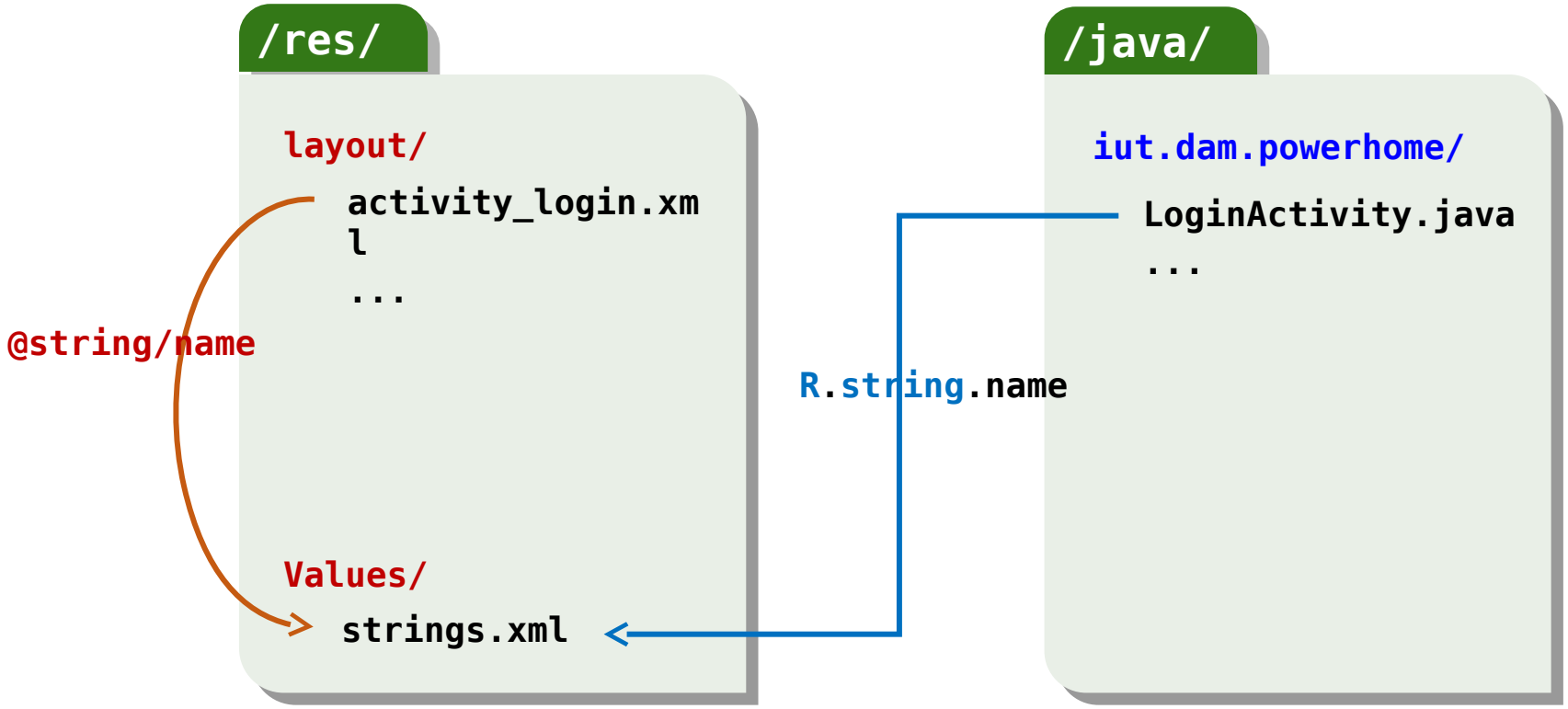
Ressources : Strings (1/2)

/res/values/strings.xml

```
<?xml version="1.0" encoding="utf8"?>
<resources>
     <string name="app_name"> PowerHome </string>
     ...
</resources>
```

- **Exploitation dans `res` (fichiers XML : layout, styles, ...) : `@string/app_name`**
- **Exploitation dans `java` (code source) : `R.string.app_name`**
  - Accès à la ressource : **`getResources().getString(R.string.app_name);`**

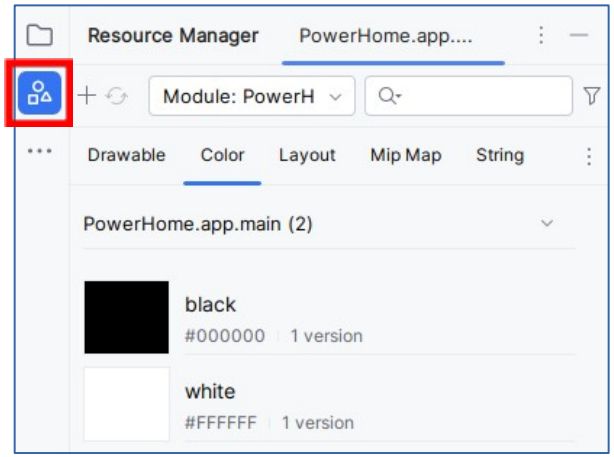
Ressources : Strings (2/2)



La classe "R"

Chaque élément défini dans le répertoire **`./res/`** impacte la classe **`R`**

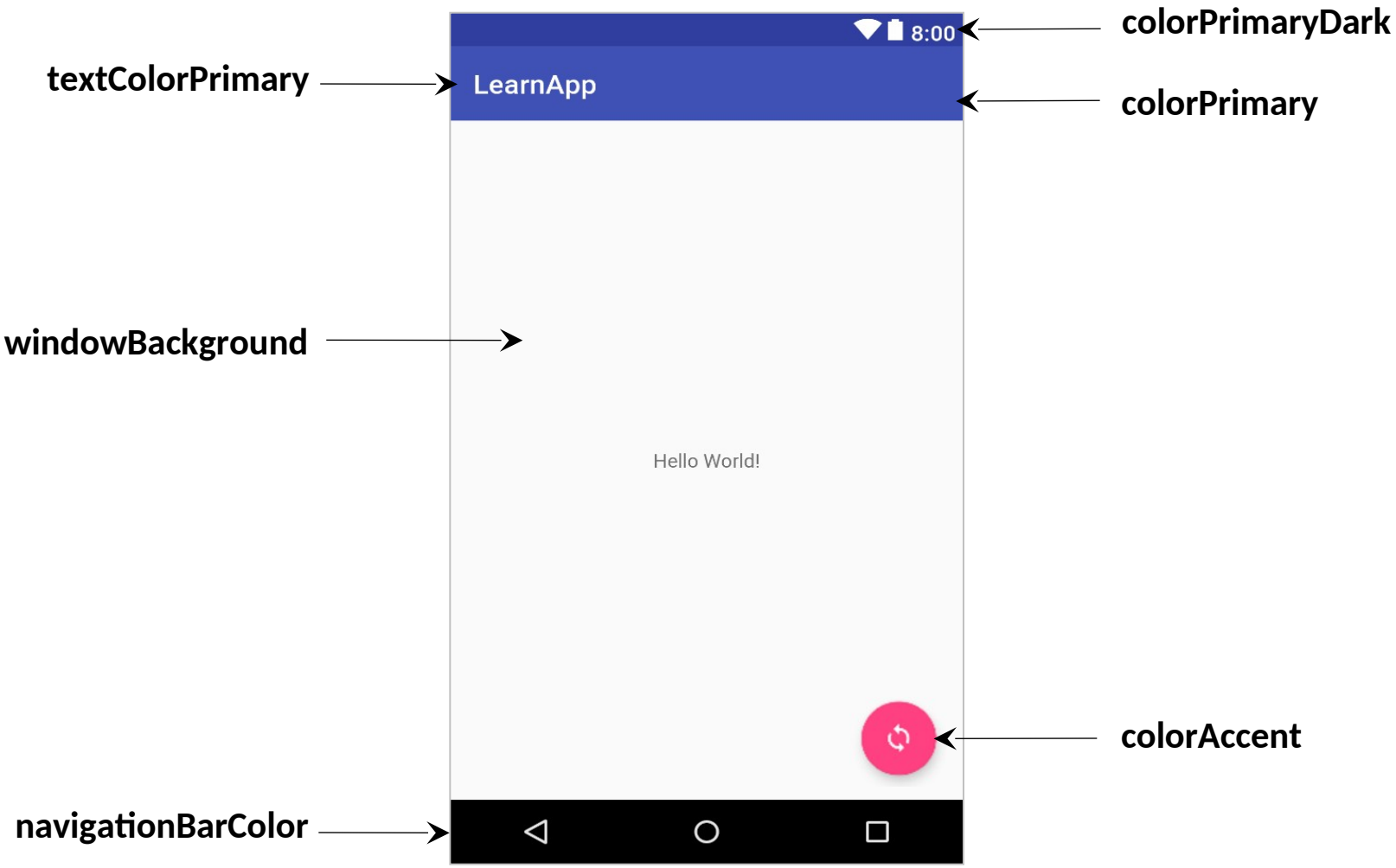
- La classe **R** référence les ressources par des constantes de type entier sur 32bits,
  - Exemple : **`0x1A34F678`**
- Depuis plusieurs versions d'**Android Gradle Plugin**, la classe R est générée directement en **bytecode** plutôt qu'en fichier .java
  - généré automatiquement
  - caché par Android studio
  - permet des builds plus rapides
- Accès aux ressources : **Tools** > **Resource Manager**



2 types de ressources via la classe R

- **user-defined :** **(java)** `R.color.vert`
  (res) `@color/vert`
- **system-defined :** **(java)** `android.R.color.holo_green_dark`
  (res) `@android:color/holo_green_dark`

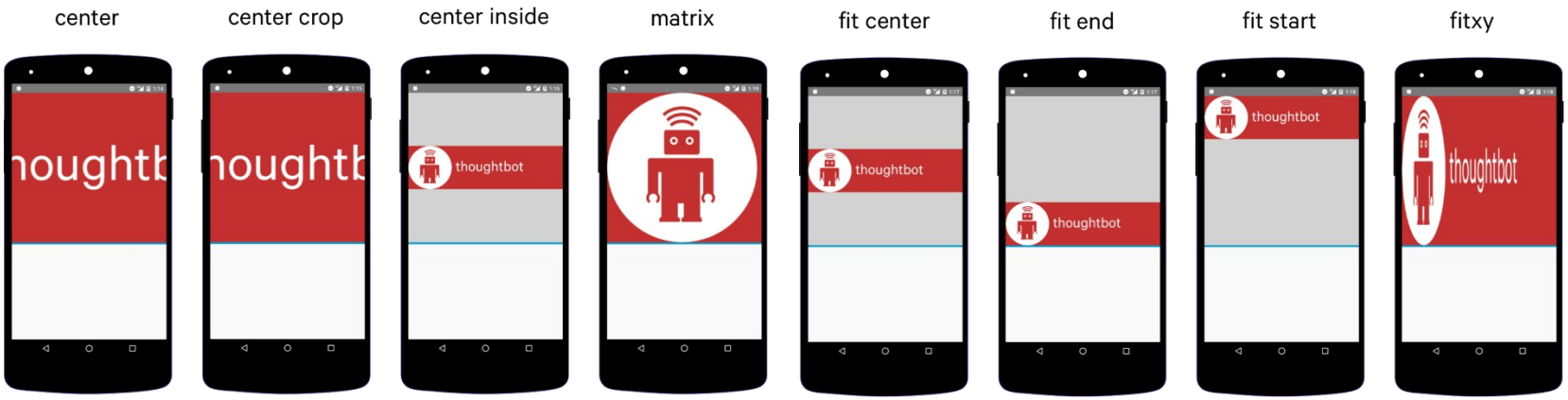
Ressources : Couleurs



Ressources : Drawables

Attributs de **ImageView**

- `src:@drawable/my_image`
- `scaleType:`

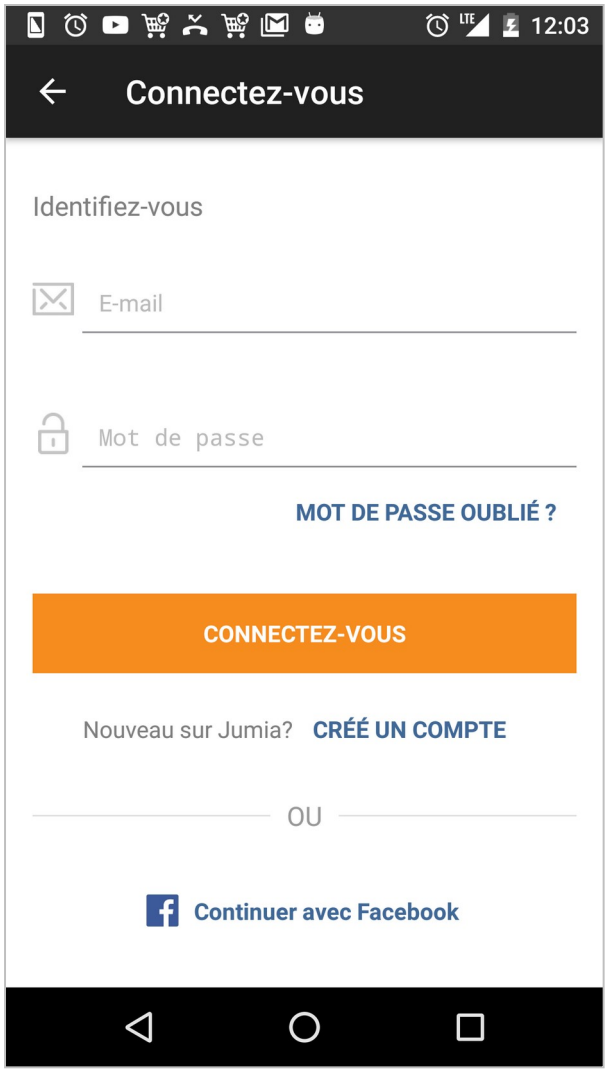


- Il faut copier l'image `my_image.png` dans `"res/drawable"`

TP1b : vue d'authentification

Créer une activité `LoginActivity`

- 2 `EditText` pour introduire
  - l'identifiant
  - le mot de passe
- 4 `Button` pour
  - Lancer l'authentification
  - S'enregistrer (création d'un nouveau compte)
  - Récupérer son mot de passe perdu
  - Se connecter avec un compte social



Application utile

Screen Alive

Lien de téléchargment :

play.google.com/store/apps/details?id=in.snapcore.screen_alive