

# Android studio

## Les premiers pas

# Terminologie

## ▶ Activity

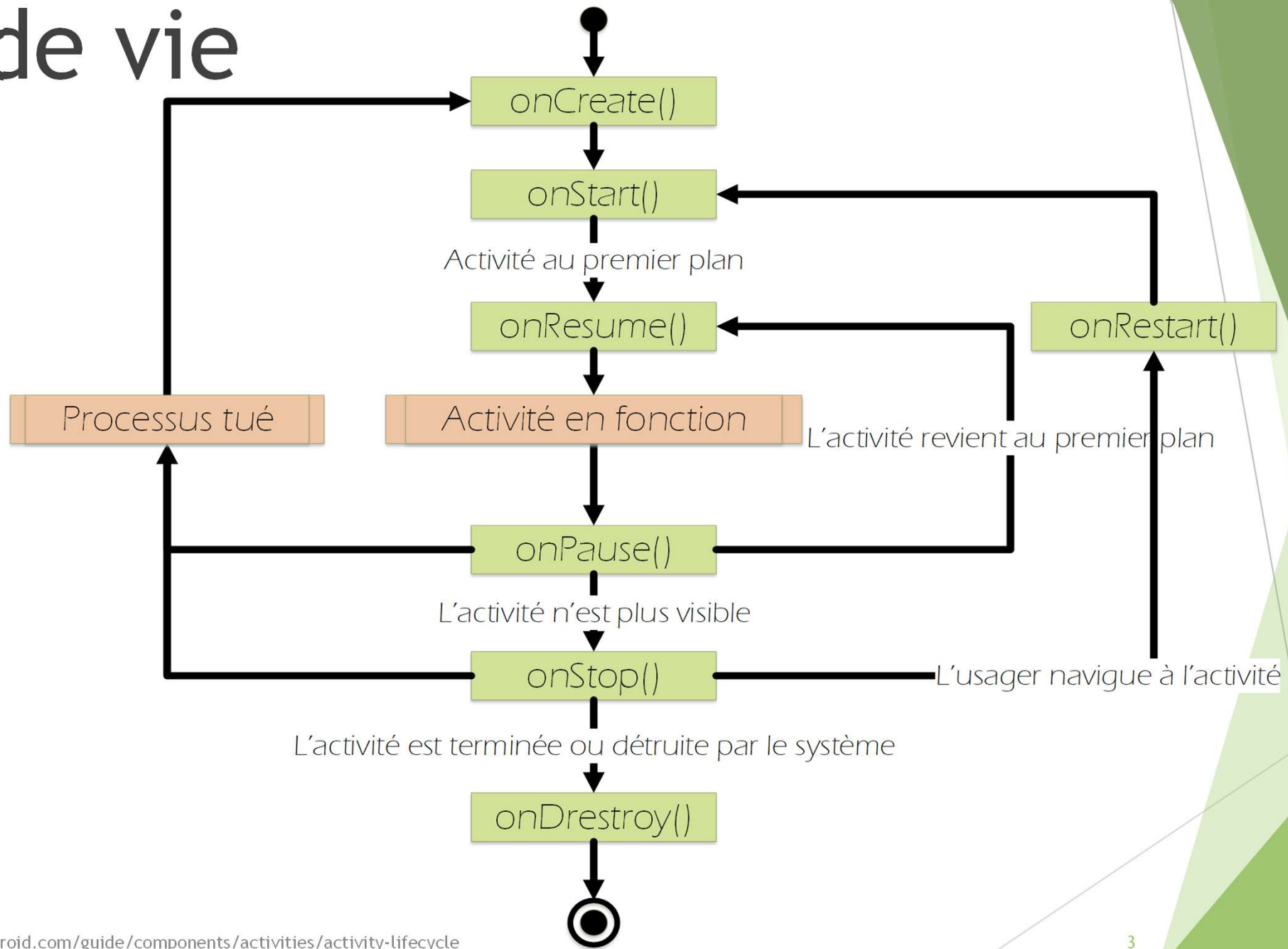
- ▶ C'est une fenêtre plein écran qui inclut l'interface et le code nécessaire pour le fonctionnement.

## ▶ Layouts

- ▶ C'est un XML qui définit l'affichage d'une interface, la disposition.



# Cycle de vie

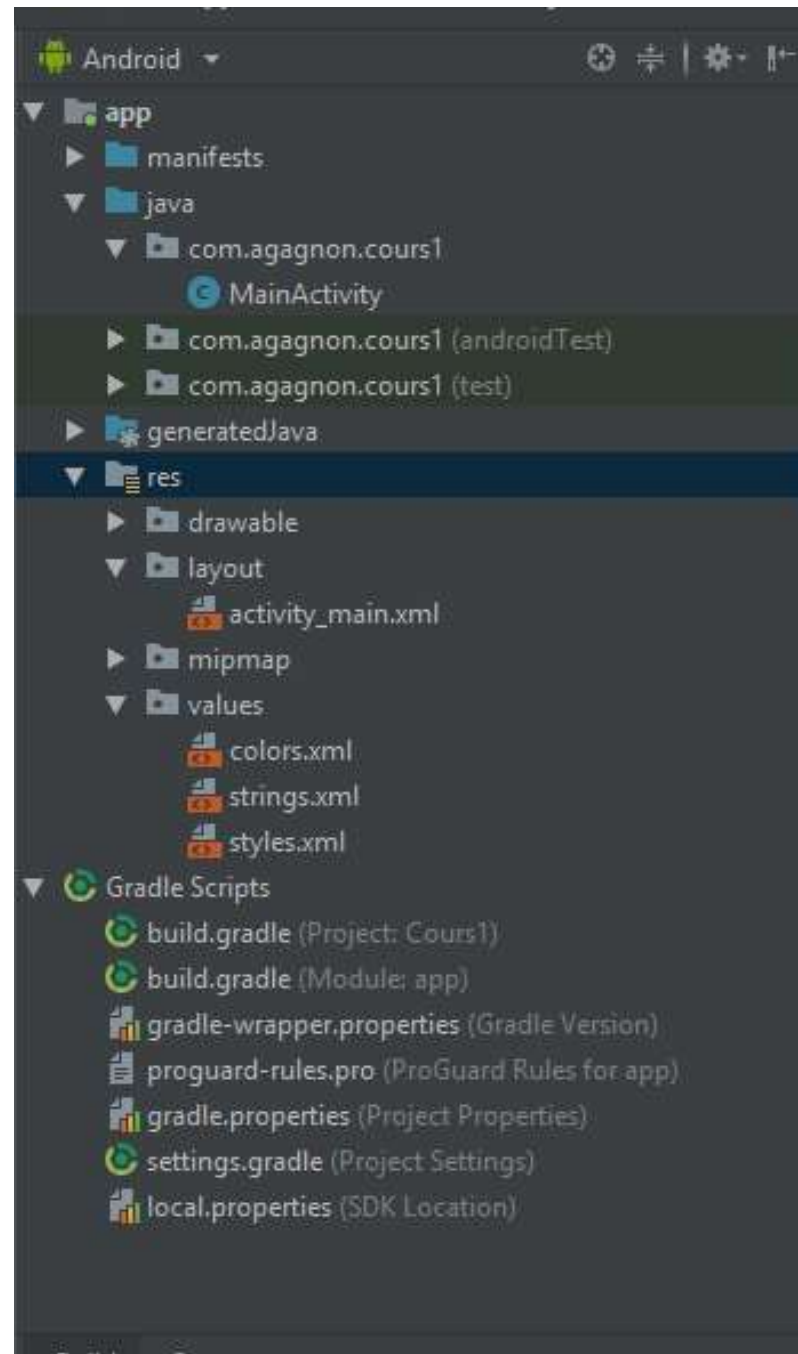


# Cycle de vie

Méthode	Description
onCreate()	Exécuté quand l'utilisateur clique sur l'icône de l'application pour la première fois.
onRestart()	L'activité repasse au premier plan.
onStart()	Chargement des données sauvegardées durant le dernier arrêt.
onResume()	Mise à jour de données possiblement modifiées.
onPause()	Exécuté chaque fois que l'utilisateur passe à une autre activité, ou bien lorsqu'il demande un finish() sur cette activité.
onStop()	Libération des ressources.
onDestroy()	Libération des ressources.

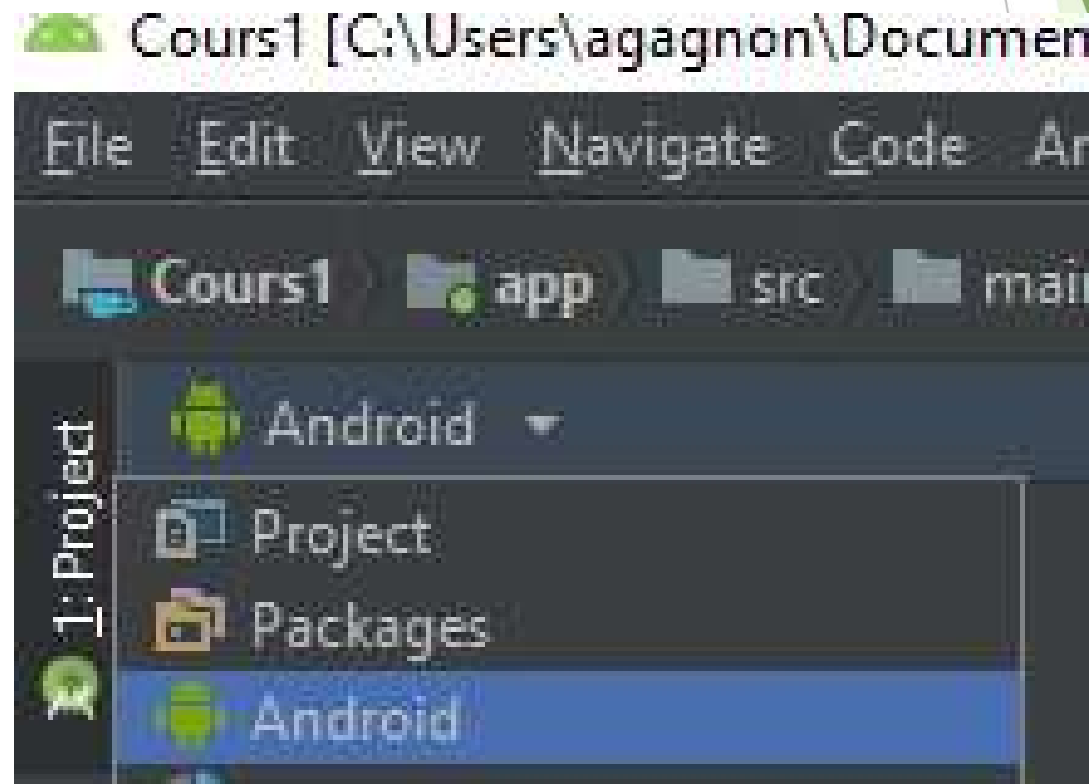


# Les répertoires



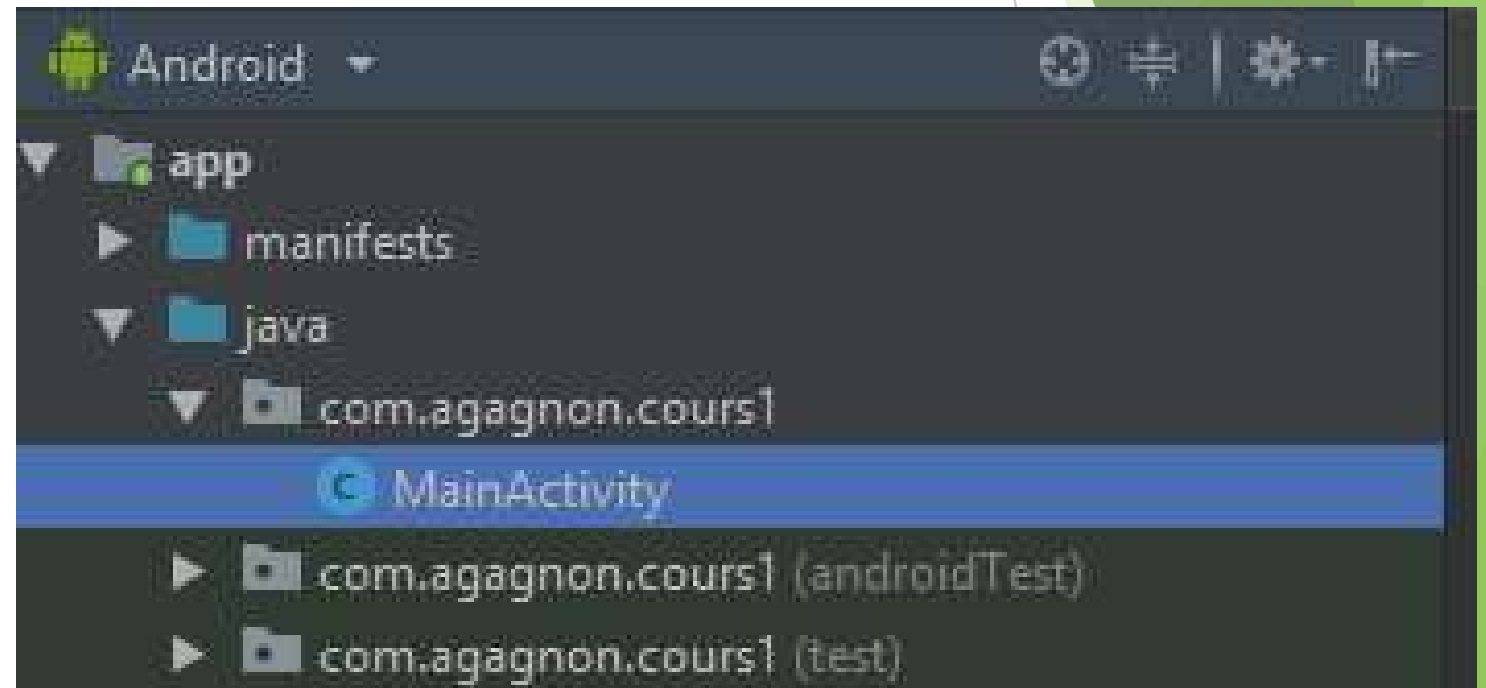
# Mode d'affichage des répertoires

Nous utiliserons  
le mode Android.



# Emplacement des fichiers de code

Ils sont situés  
dans  
`app/java/nomDe  
LApplication`



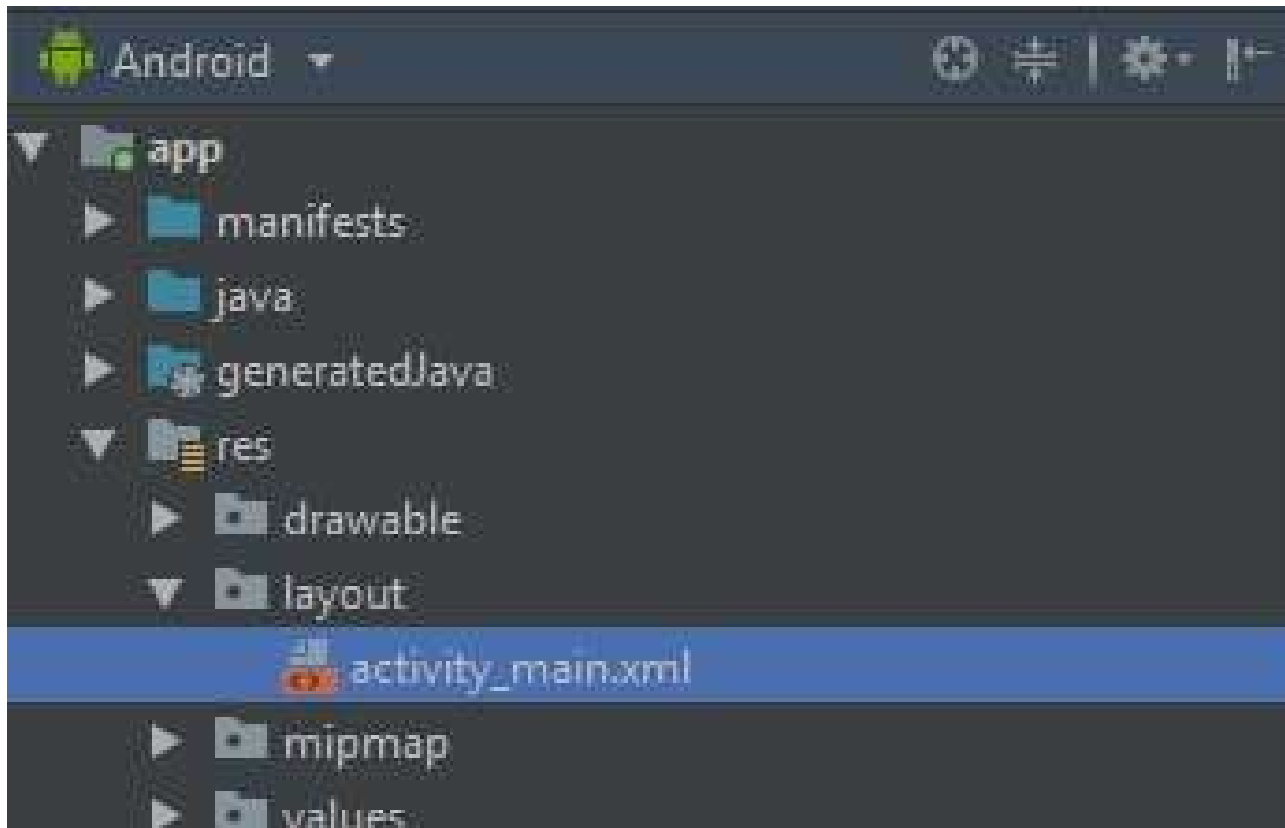
# Les layouts

Un layout est ce qui permet de générer une interface avec des « Views ». C'est en résumé une manière générique de définir comment s'affichera un écran. Ceux-ci sont sauvegardés en format XML.

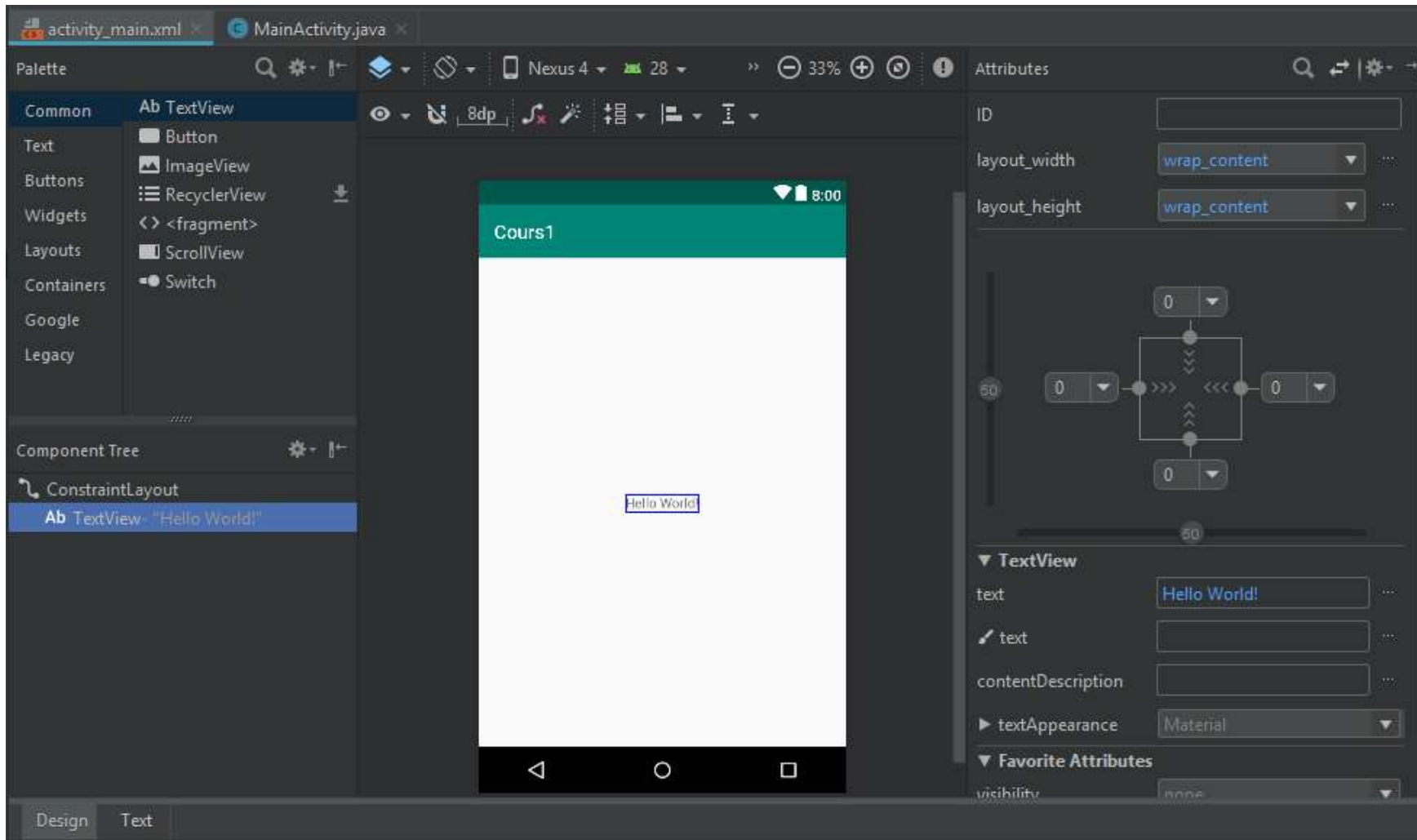




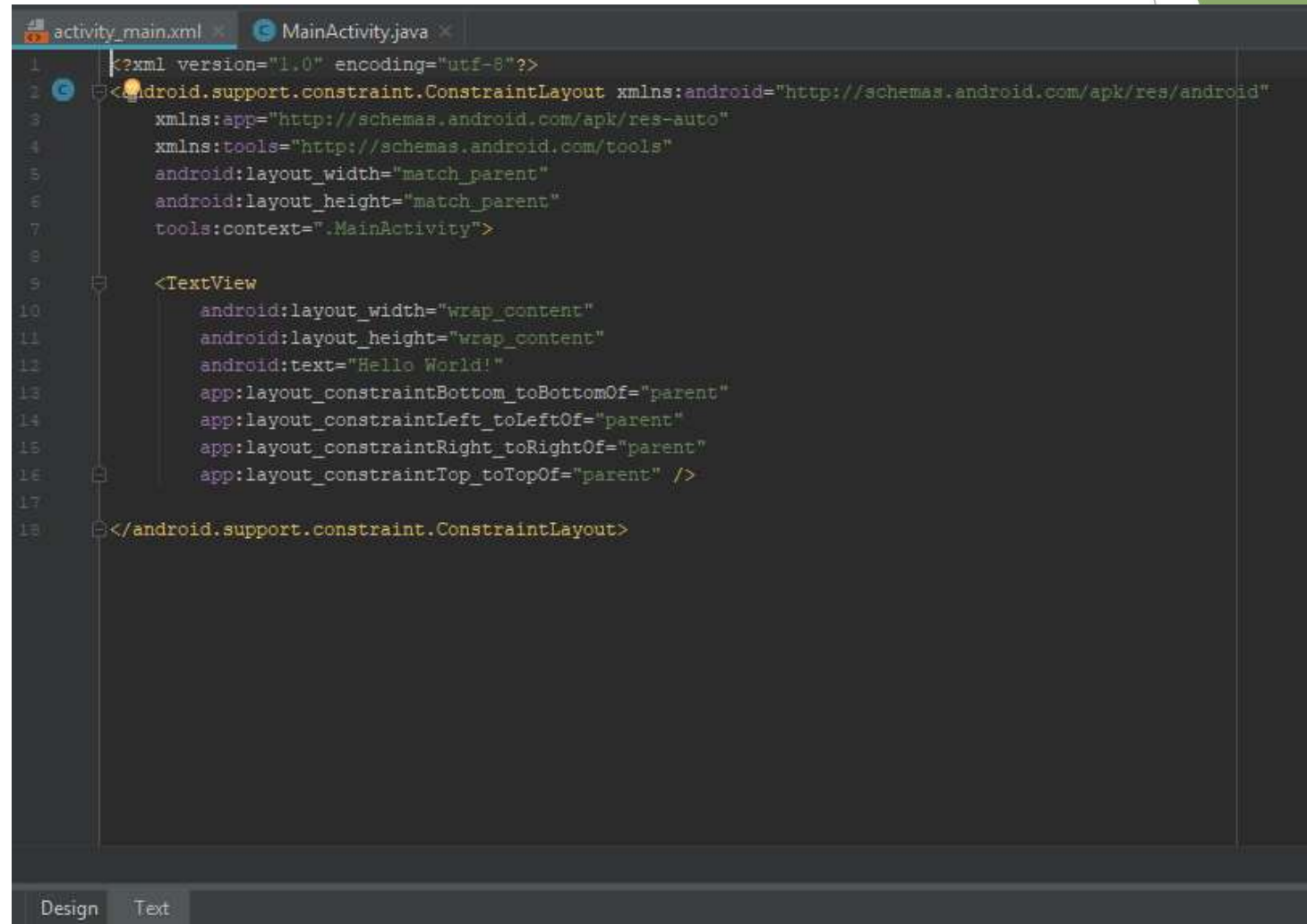
# Où se retrouvent les layouts



# Fenêtre de modification de layout



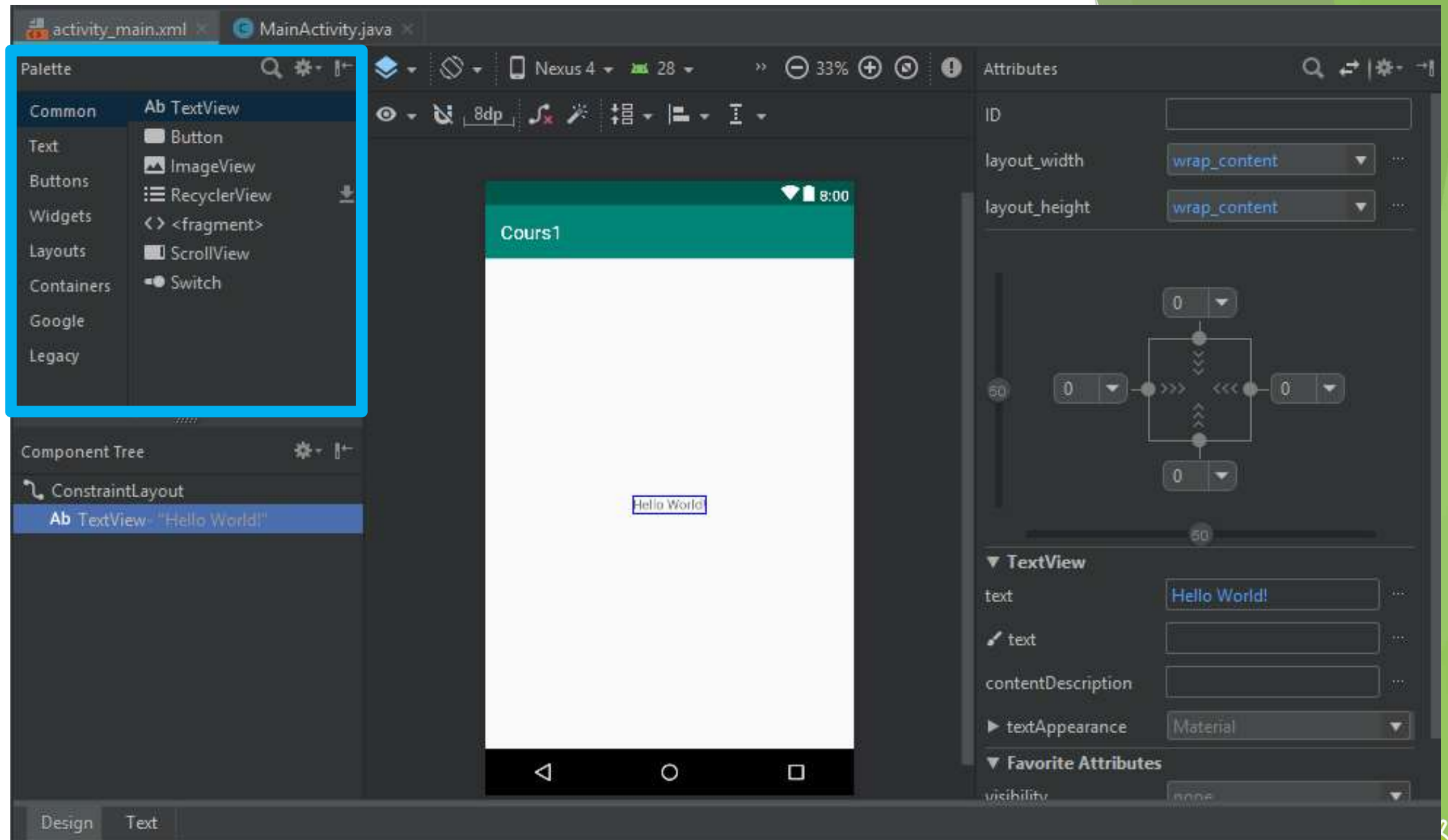
# Fenêtre de modification de layout(XML)



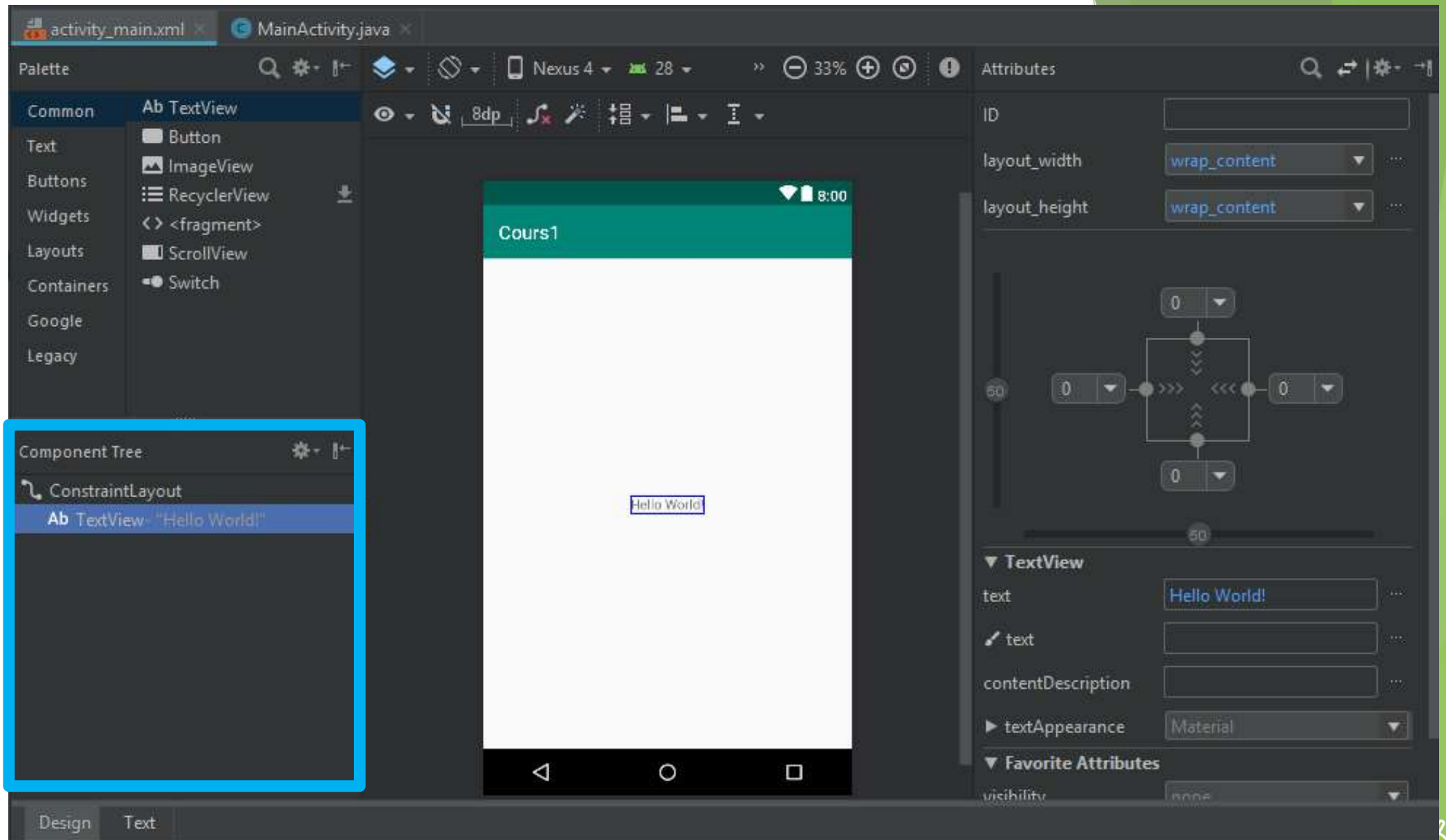
```
1 <?xml version="1.0" encoding="utf-8"?>
2 <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     tools:context=".MainActivity">
8
9     <TextView
10         android:layout_width="wrap_content"
11         android:layout_height="wrap_content"
12         android:text="Hello World!"
13         app:layout_constraintBottom_toBottomOf="parent"
14         app:layout_constraintLeft_toLeftOf="parent"
15         app:layout_constraintRight_toRightOf="parent"
16         app:layout_constraintTop_toTopOf="parent" />
17
18 </android.support.constraint.ConstraintLayout>
```



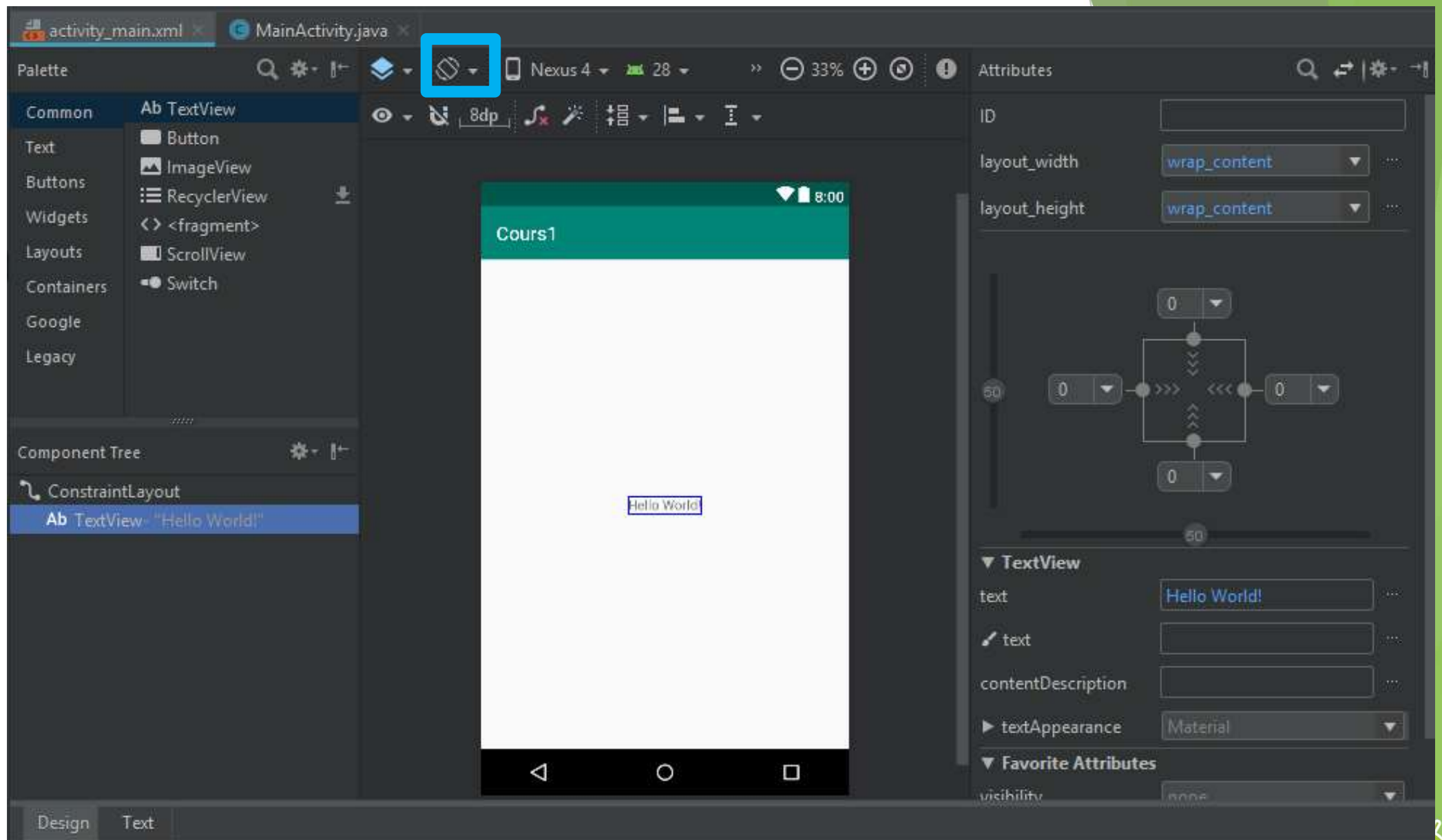
Section qui permet d'ajouter des views



Arborescence  
du layout.  
Des views  
peuvent  
contenir des  
views.

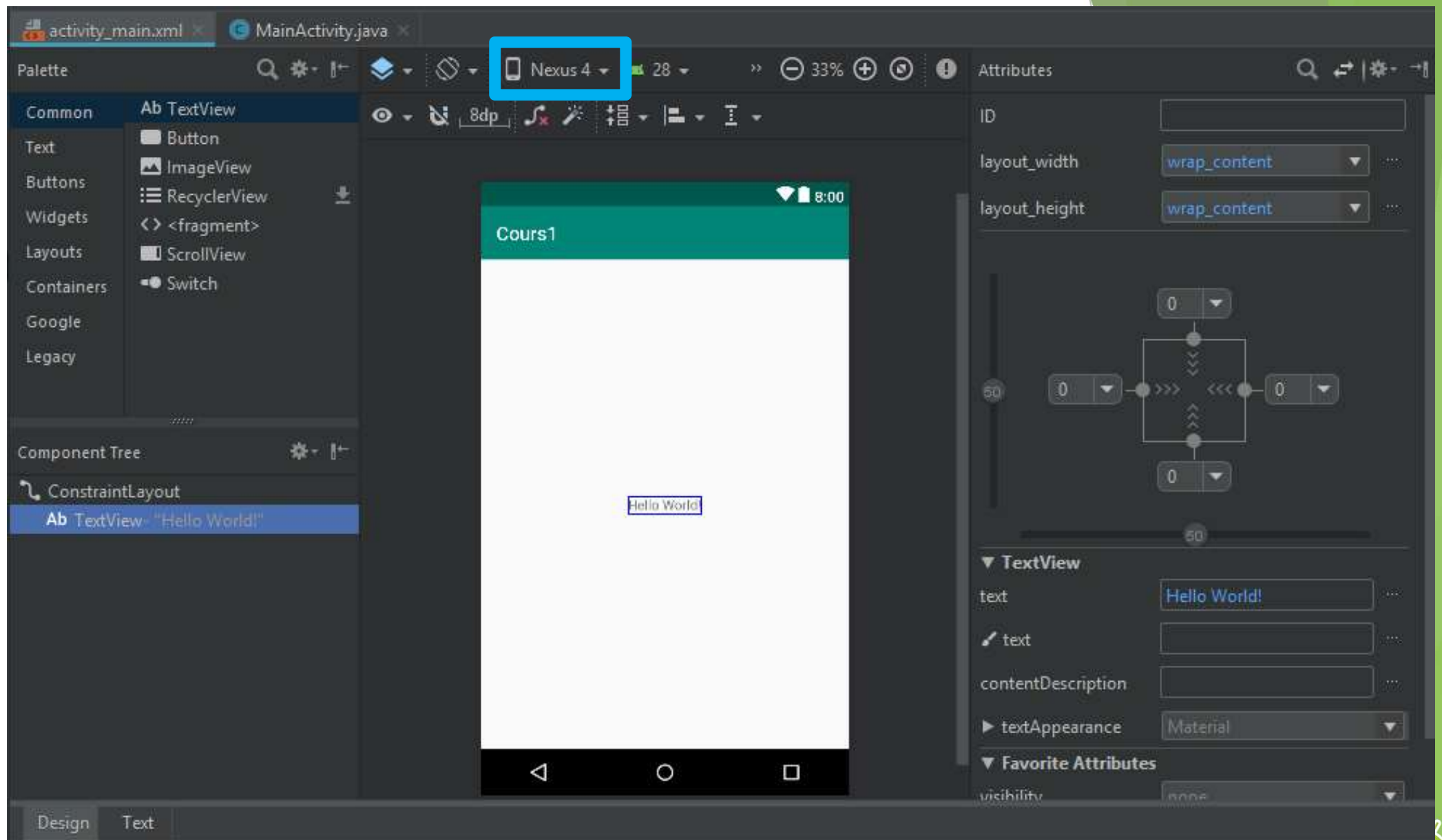


Changer  
l'orientation  
de l'appareil

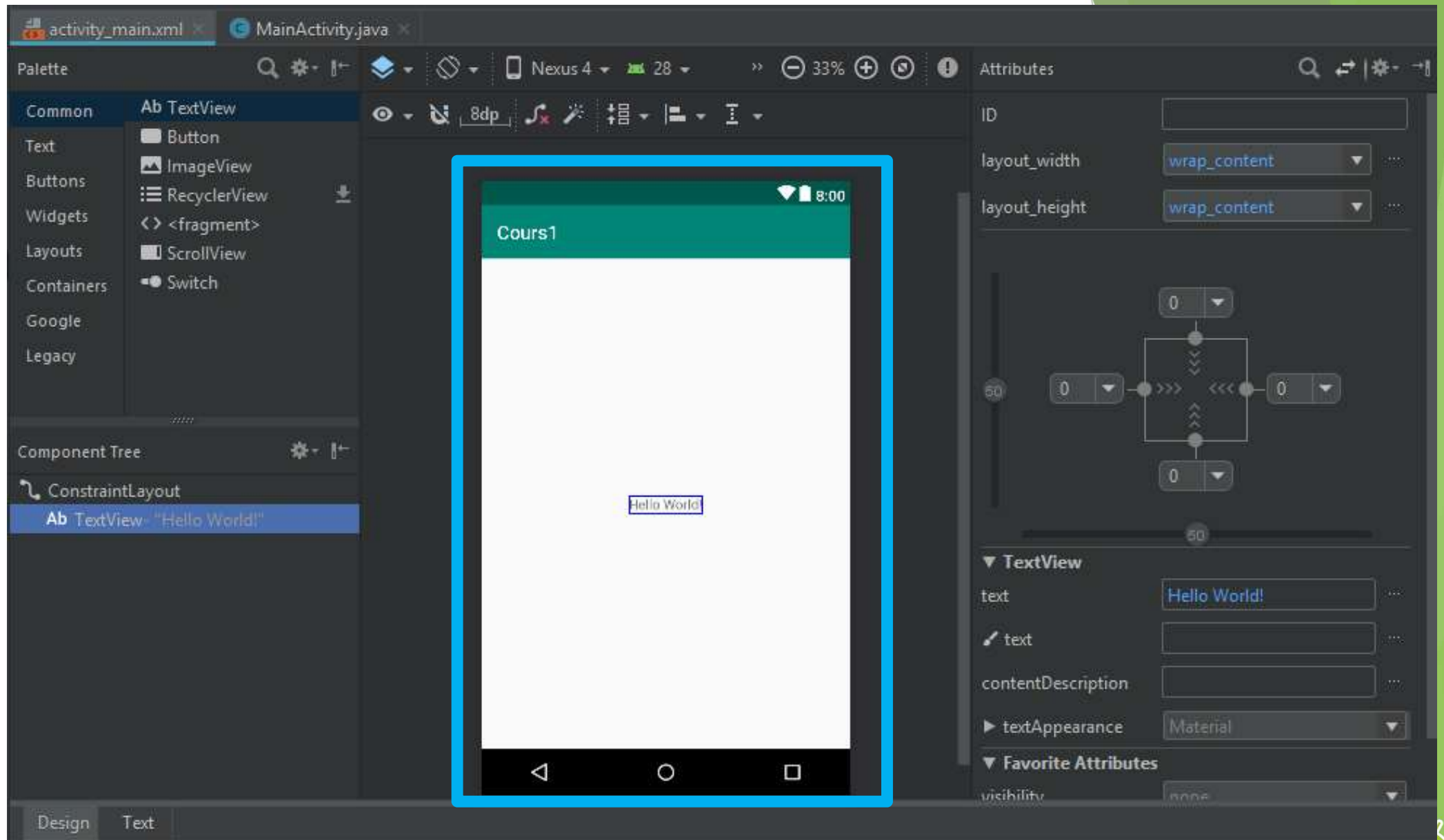




Changer  
d'appareil

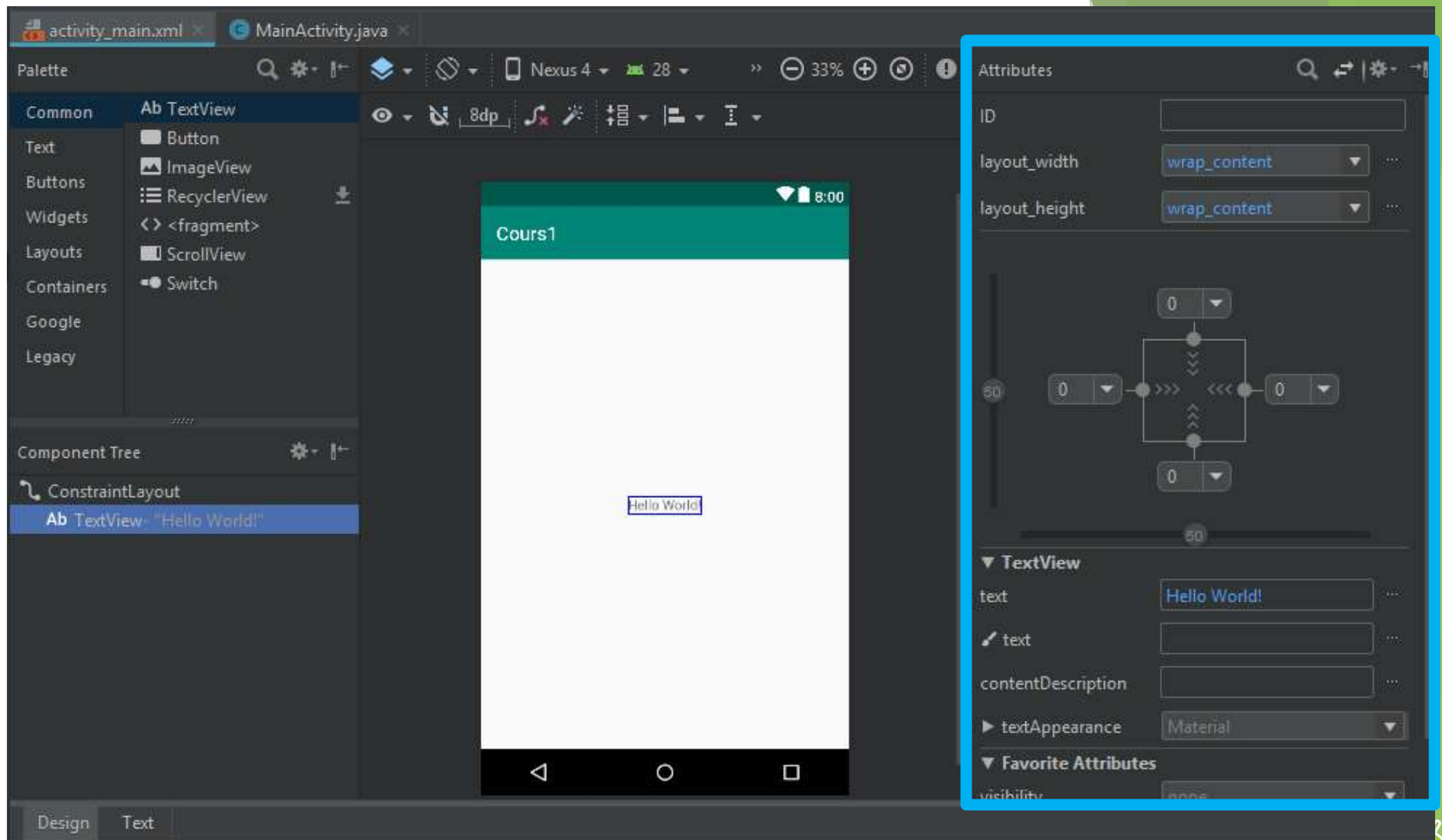


Aperçu du  
résultat





## Propriétés de la view sélectionnée



# Modification d'un layout

En sélectionnant un layout il est possible de le modifier. Android studio offre une interface visuel pour modifier les layouts mais il est également possible de modifier un layout directement dans son XML.



# Modification d'un layout

Bien évidemment une modification d'un layout directement dans l'interface visuelle modifiera le XML.



# Modification d'un layout

Vous remarquerez que les exemples d'internet sont généralement en XML.



# Unité de mesure

- ▶ dp
- ▶ sp
- ▶ pt
- ▶ px
- ▶ mm
- ▶ in



# dp, density-independent Pixels

Une taille indépendante de la résolution(DPI). Cela permet un affichage uniforme sur plusieurs appareils.



# sp, Scale-Independent pixels (meilleur pour le texte)

Une taille indépendante de la résolution(DPI). Cela permet un affichage uniforme sur plusieurs appareils. Toutefois, la taille sera modifiée en fonction des préférences de tailles de polices de l'utilisateur.



pt

1/72 de pouce.





# px

1 pixel. Une meilleure résolution aura des éléments plus petits qu'une résolution plus basse.



mm

1 millimètre



in

Unité de mesure impériale  
représentant « Inch » (le pouce).



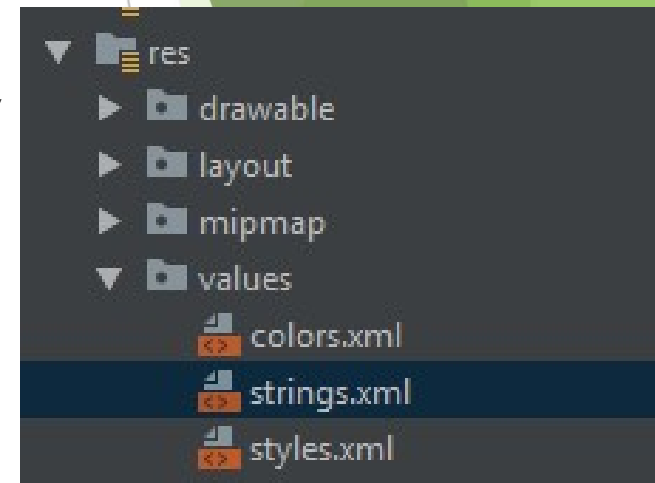
# Dans le cadre du cours

Nous utiliserons SP et DP autant que possible et je demande d'en faire de même dans votre projet.



# Les fichiers de ressources

Permettent de contenir les images, mais aussi des valeurs à un endroit commun pour éviter les valeurs magiques dans le code. Dans le cadre du cours, nous utiliserons rarement ces fichiers pour un gain de temps, mais il est important de comprendre qu'il s'agit d'une très mauvaise pratique. Dans un vrai projet, chaque valeur devrait être dans l'un de ces fichiers.



# Le manifest

- ▶ Permet de définir les permissions
- ▶ Permet de définir les activités ainsi que l'activité initiale à l'ouverture de l'application
- ▶ Exigence matérielle et logiciel pour le fonctionnement de l'application
- ▶ Et autre...



# Le gradle

Le gradle est un système de construction de projet. Ce système n'est pas exclusif au projet Android. Sans le gradle on devrait se faire un script permet de passer plusieurs commandes pour obtenir un APK en incluant nos images, vidéo, sons ainsi que dépendance externe.



# Précision Gradle

Votre projet contient 2 gradles.  
Nous modifierons habituellement  
celui du module et pas celui du  
projet.

