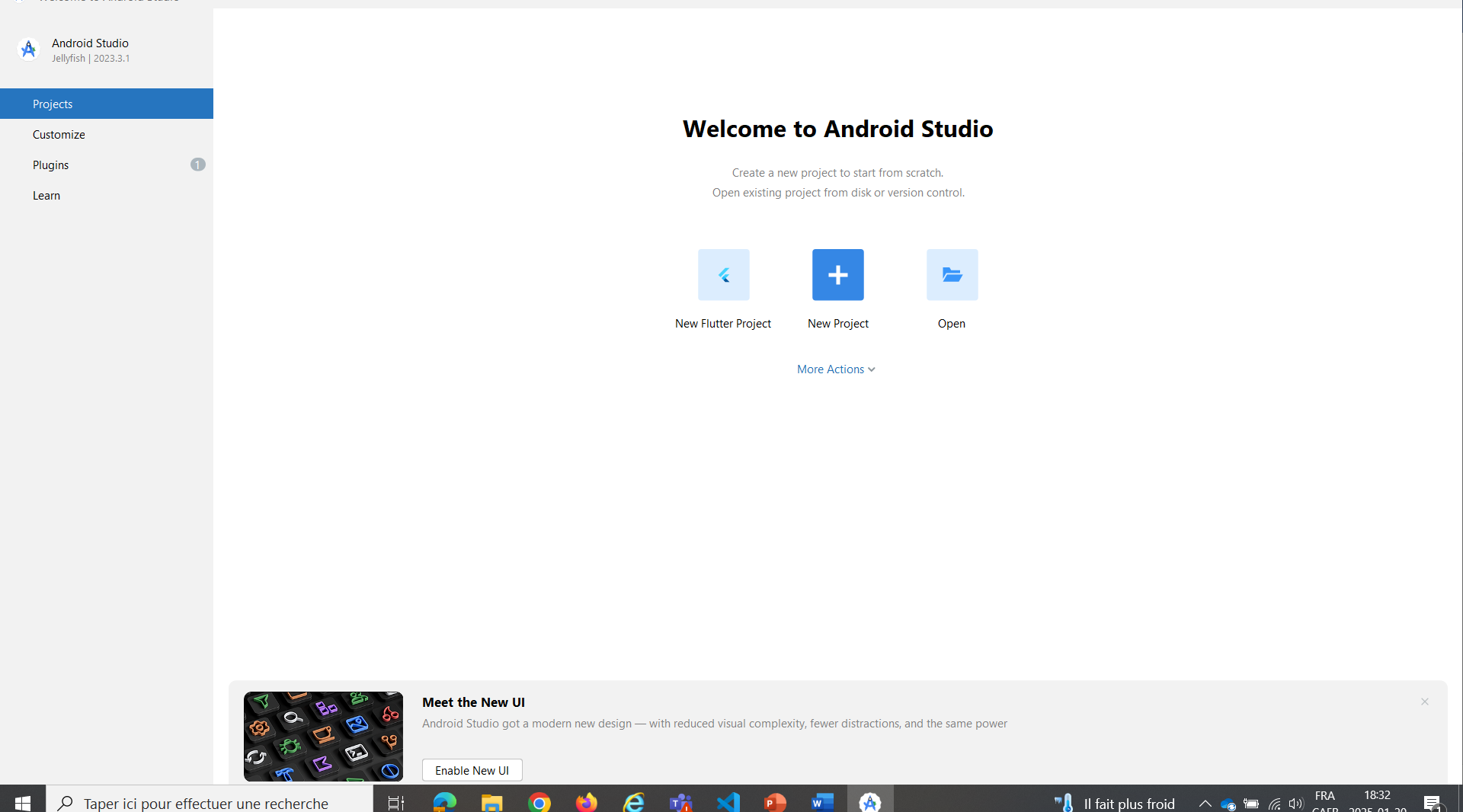
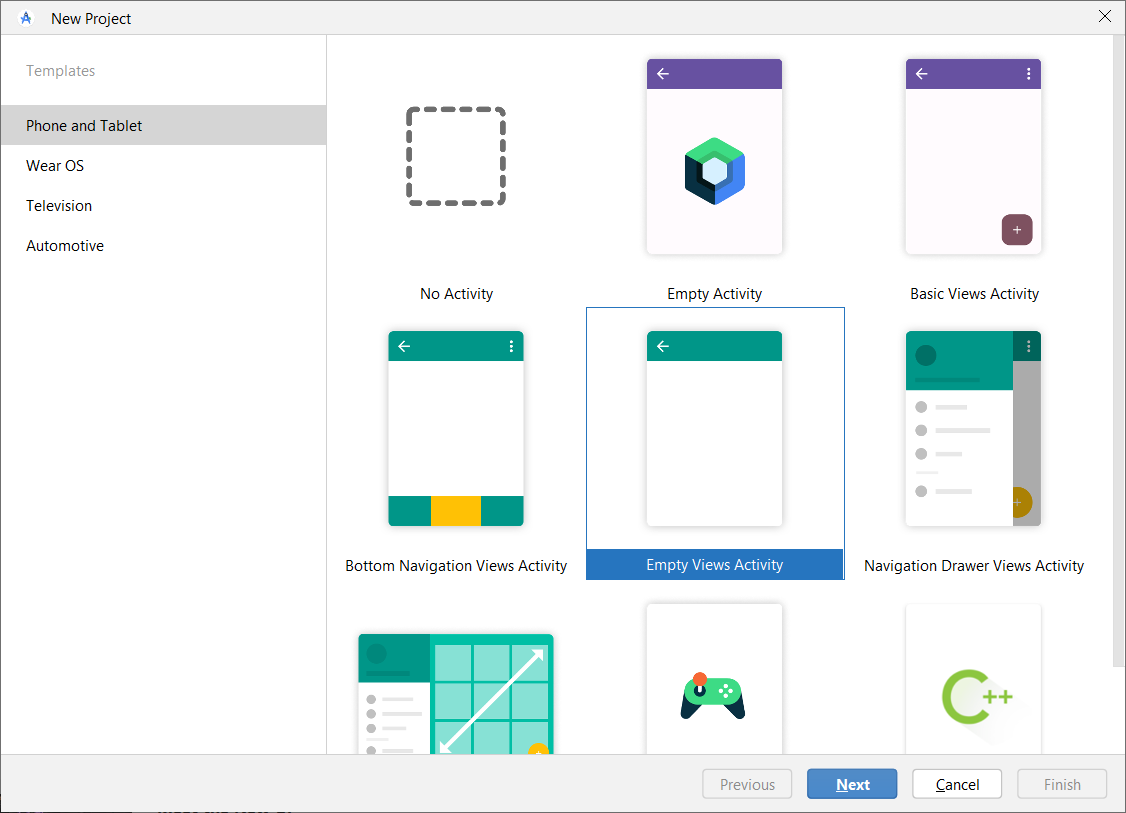
# Annexe 1 –AndroidStudio – premier projet

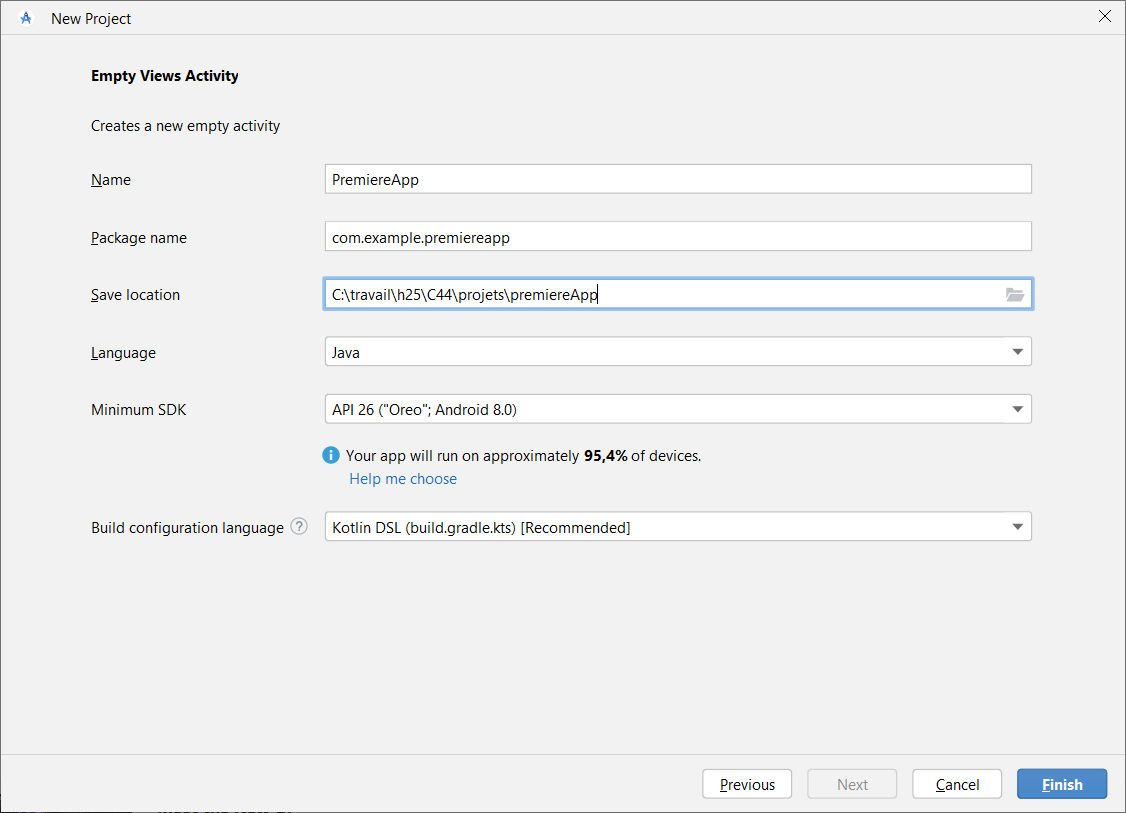
1. Démarrer Android Studio ( cette session : )



Faire “New Project

* Qu’est ce qu’une Activité ?
* 2. Choisissez « Empty Activity » et Faites Next 🡪



3 

**Package** : doit avoir au moins 2 parties séparées par un point, **nom doit être unique** en vue d’une publication ( nom de votre compagnie ) ex: eric.labonte.international.helloworld

Name : Nom du projet représentant l’Application. Ce nom sera utilisé pour nommer l’app sur le téléphone

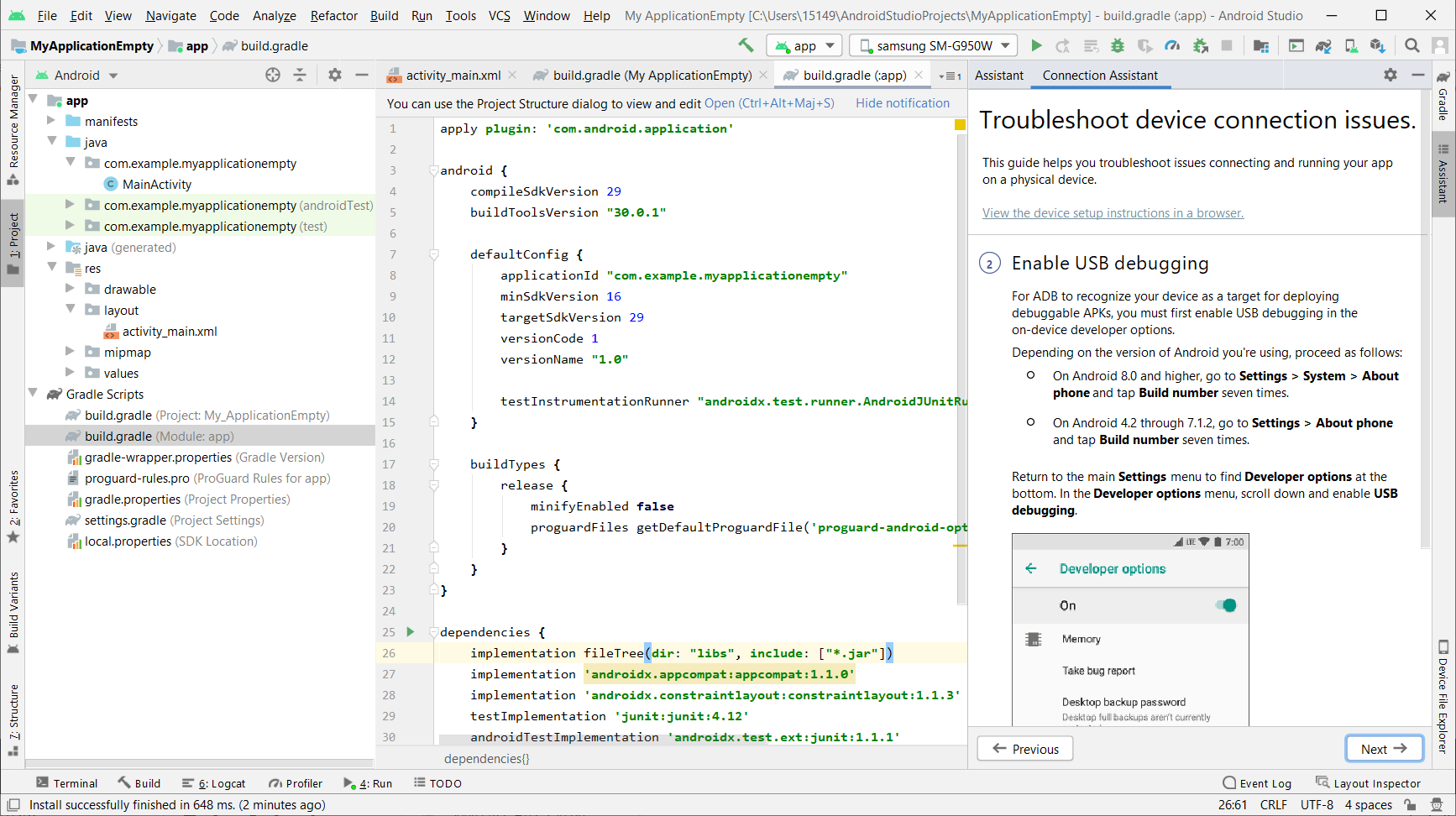
Minimum API level : **On va choisir 26 comme convention locale**, même si on le logiciel compilera avec la version plus récente installée ( cette session 34) .

Les API correspondent aux différentes versions d’Android ( bien que numérotées différemment ) :

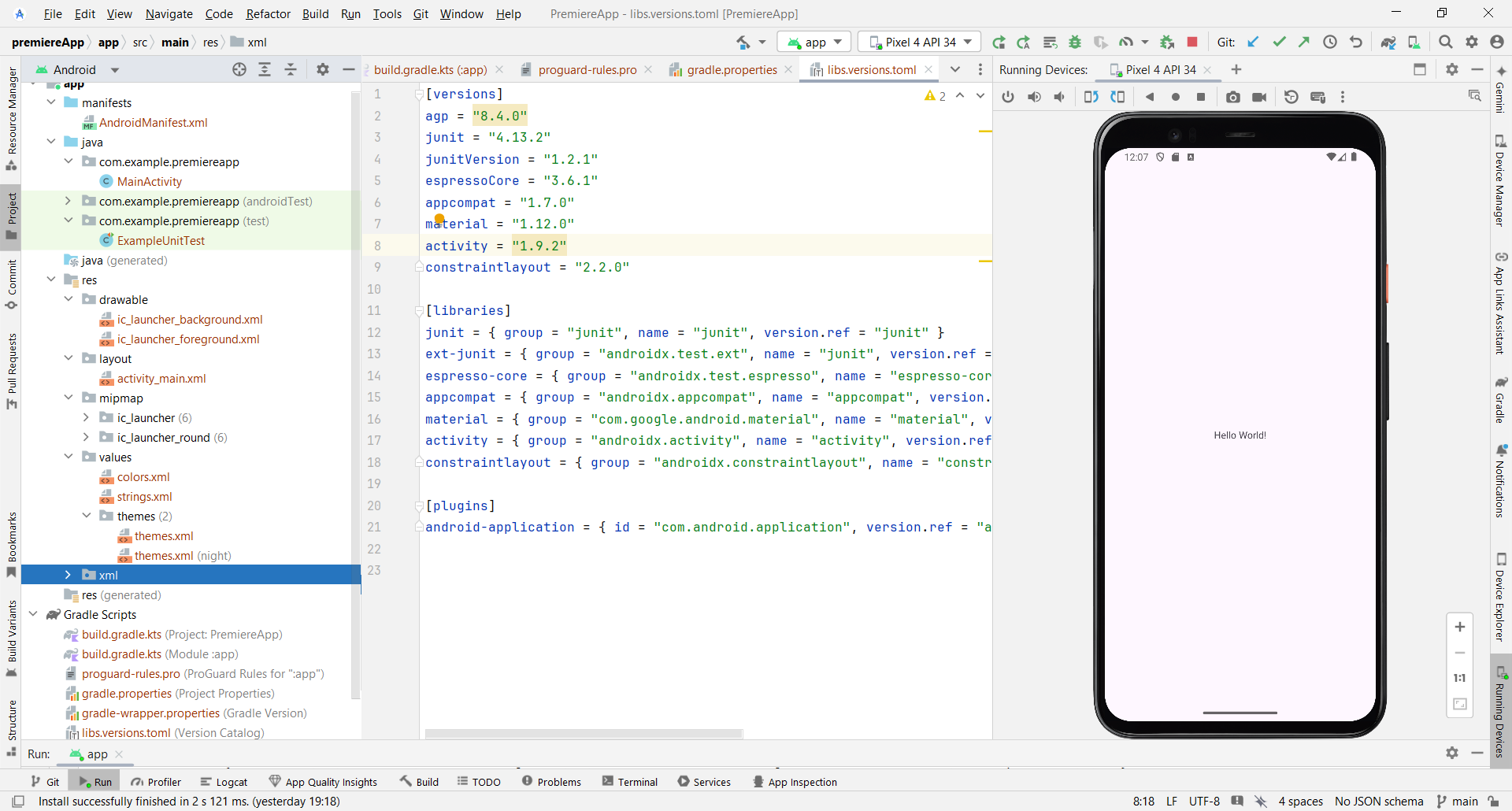
|  |  |  |
| --- | --- | --- |
| Nom de la version | No. de la version | No. de l’API correspondante |
| Jelly Bean | 4.1 – 4.3.1 | 16 – 18 |
| KitKat | 4.4 – 4.4.4 | 19 – 20 |
| Lollipop | 5.0 – 5.1.1 | 21 – 22 |
| Marshmallow | 6.0 – 6.0.1 | 23 |
| Nougat | 7.0 – 7.1.2 | 24 – 25 |
| Oreo | 8.0 – 8.1 | 26 – 27 |
| Pie | 9 | 28 |
| Android 10 / Q / Quince Tart | 10 | 29 |
| Android 11 / R / RedVelvet Cake | 11 | 30 |
| Android 12 / S / Snow Cone | 12 | 31-32 |
| Android 13 / Tiramisu | 13 | 33 |
| Android 14 / Upside Down Cake | 14 | 34 |
| Android 15 / Vanilla Ice Cream | 15 | 35 |
| Android 16 / Baklava | 16 | 36 |

## Intéressant : utiliser son propre téléphone plutôt qu’un émulateur…

* Nous utiliserons l’API 26, lors de la création des projets, vous pouvez utiliser une version plus basse correspondant à votre version d’Android ( vous pouvez la modifiez en cours de route en faisant Menu File 🡪 Project Structure 🡪 Modules 🡪onglet Default Config 🡪Min SDK Version )
* Pour que vous puissiez utiliser votre téléphone, vous devez rendre utilisable le « USB debugging » en ayant accès aux options de développement de votre téléphone Android :



## Types de fichiers dans un projet



Cliquez ou appuyez ici pour entrer du texte.

Cliquez ou appuyez ici pour entrer du texte.

Cliquez ou appuyez ici pour entrer du texte.

Cliquez ou appuyez ici pour entrer du texte.

Cliquez ou appuyez ici pour entrer du texte.

Cliquez ou appuyez ici pour entrer du texte.

Cliquez ou appuyez ici pour entrer du texte.

Cliquez ou appuyez ici pour entrer du texte.

Cliquez ou appuyez ici pour entrer du texte.

## Exemple 0 : Faire rouler l’app avec HelloWorld

**IMPORTANT ( HIVER 2025 ) : On doit modifier une dépendance ( import ) dans chacun des projets qu’on fera.**

**Fichier libs.versions.toml 🡪 activity 🡪 changer 1.10.0 pour 1.9.2 🡪 Faire Sync Now**

## Exemple 1: dessiner une interface utilisateur relative à une application de transaction bancaire.

1. Pour dessiner l’interface, on utilise le fichier : activity\_main.xml
2. On peut diviser l’Activité en plusieurs sections à l’aide de conteneurs appelés Layouts.

Quel Layout est présent par défaut ? constraintLayout

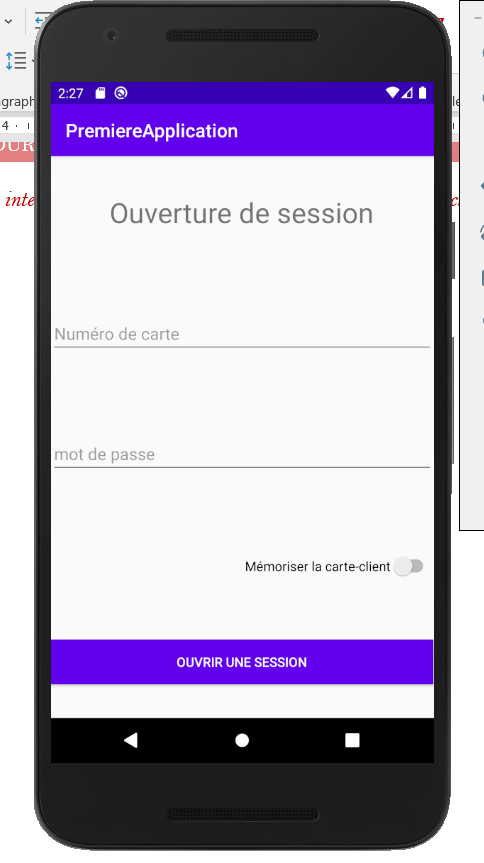
1. Les composants ( widgets ) dont on aura besoin :

TextView : Texte non-éditable

EditText ( texte ) : éditable

Switch : interrupteur à 2 valeurs

Button : bouton

  
4. Propriétés importantes ( palette à droite ) :

id : identifiant du composant

text : changer le text de la composante

background : changer la couleur de fond

Oops…on ne peut pas changer la couleur de fond du bouton avec la propriété background mais avec background\_tint car le thème employé par l’application est un thème de type Material 🡪 on peut également modifier le thème dans le fichier themes.xml ( genre de css )

Pour plus d’infos :

<https://material.io/design/color/the-color-system.html#color-usage-and-palettes>

[https://m2.material.io/design/color/applying-color-to-ui.html#buttons-chips-and-selection-controls](https://m2.material.io/design/color/applying-color-to-ui.html%23buttons-chips-and-selection-controls)

Comment on doit ajouter les composants dans un ConstraintLayout ( on pratique avec le TextView « Ouverture de session » ) :

* La position de chaque widget est déterminée par des contraintes
* Les contraintes d’un widget sont soit par rapport au parent ( objet ConstraintLayout ) ou par rapport aux autres widgets.
* Au minimum, chaque widget doit avoir une contrainte verticale et une contrainte horizontale ( sinon les composantes s’accumuleront dans le coin supérieur gauche, même si ça semble correct dans l’onglet Design )

**Faire rouler l’app**

En faisant RUN dans l’interface principale d’AndroidStudio, vous devriez voir votre première Application

**Pratique :**

* Positionnement : au moins deux contraintes
* Centrer : 2 contraintes latérales équivalentes
  + Bias : calibrer l’orientation avec la roulette
* Aligner :
* Guides :

Barrière : <https://constraintlayout.com/basics/barriers.html>

* Chaînes :

<https://constraintlayout.com/basics/create_chains.html#:~:text=Chains%20are%20a%20specific%20kind,that%2C%20as%20we%20shall%20see>

* **Dimensions des widgets : match\_constraint ou wrap\_content**

**On place les autres widgets / composantes et on fait exécuter le programme sur l’émulateur / votre téléphone.**

**Notes supplémentaires :**

Taille du texte : TextSize en sp (scale independant pixel)

Hint : suggestion