# Première application Android

## Objectifs

Se familiariser avec l’environnement Android Studio.

Utiliser Java pour développer une interface graphique.

Développement d’application mobile pour Android.

## Fonctionnalités de l’application

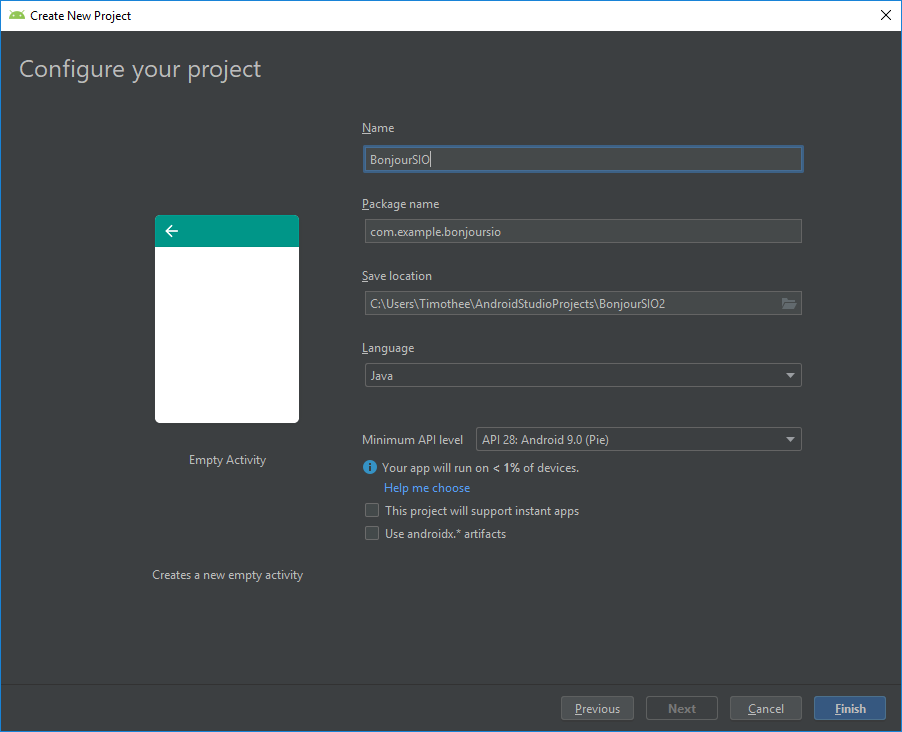
L’utilisateur doit saisir une matière dans un champ texte, puis cliquer sur un bouton qui renvoie une page de documentation associée à la matière.

## Contenus

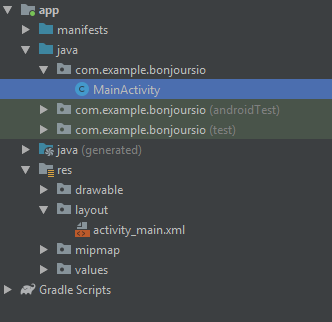
Environnement Android Studio – IHM – Java – outil de build Gradle (packaging et dépendances) – layouts – gestion des évènements – listeners - Intents

**Partie 0 : création d’un nouveau projet Android Studio**

Définissez une nouvelle activité « vide » (empty activity)

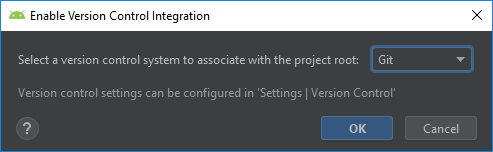


Pensez bien à spécifier Java comme langage, étant donné que Kotlin est dorénavant le langage par défaut d’Android Studio.



Mise en place du versionning : dans le menu VCS, choisir la mise en place du versionning avec GIT (enable … Git).

VCS 🡪 Enable version control integration puis choisir Git



**Partie 1 : création de l’interface graphique**

**Zoom cours :** principes de l’interface graphique Android

L’interface utilisateur est construite au moyen d’une hiérarchie imbriquant des ***layouts*** et des ***widgets***.

<https://developer.android.com/guide/topics/ui/declaring-layout>

<https://developer.android.com/training/basics/firstapp/building-ui>

Etape 1 : modification des défauts et ajout d’attributs

Se placer en mode ***xml*** :

Le fichier sur lequel on travaille est **activity\_main.xml**

Vous pouvez garder le ConstraintLayout par défaut. C’est un layout récent mais surpuissant. Si vous faites des tutoriels sur Internet, vous verrez souvent l’utilisation de LinearLayout : n’hésitez pas à le remplacer par un ConstraintLayout.

Dans la définition du TextView déjà existant, ajoutez une marge (layout\_margin) et de l’espacement (padding).

L’unité préférentielle est le **dp** : <https://developer.android.com/guide/topics/resources/more-resources#Dimension> ou <http://www.nicolasaguenot.com/nicoandco/index.php?post/2011/11/30/Les-diff%C3%A9rentes-unit%C3%A9s-de-mesures-Android>

Etape 2 : ajout d’un composant **EditText**

Sous le TextView, ajoutez un composant EditText : commencez à saisir le marqueur de démarrage de balise < puis un e majuscule E et AndroidStudio va vous proposer automatiquement la bonne balise.

Choisissez wrap\_content pour les 2 layouts.

Enfin ajoutez un attribut « hint » qui doit guider l’utilisateur.

Passez en mode Design 🡪 Blueprint et ajoutez une contrainte en haut top) entre composants.

Etape 3 : ajout d’un composant Bouton

A ajouter sous le EditText.

Ajoutez/modifiez du texte (par exemple « Soumettre » au lieu de « Button »).

**Exercice 1 :**

Repasser en mode design. Observez le Component Tree : vous devez voir des alertes ?

Effectuez les corrections nécessaires pour ne plus avoir d’alerte.

<https://developer.android.com/guide/topics/resources/string-resource>

**Partie 2 : interaction dans le code avec les éléments de l’interface graphique**

**Zoom cours**

Dans android, pour accéder aux éléments de l’interface graphique, il faut qu’ils aient un identifiant.

L’identifiant est spécifié au moyen de l’attribut ***id*** et d’une valeur associée commençant par ***@+id***.

Exemple :

<**TextView  
 android:id="@+id/sio\_texte"**

Documentation officielle : <https://developer.android.com/reference/android/app/Activity>

La syntaxe des identifiants de widget android peut sembler bizarre au début mais elle est très pratique à l’usage pour identifier facilement ces identifiants.

**Question 1** : ouvrez le fichier MainActivity.java

1. Quels identifiants sont référencés dans ce fichier ?
2. Quels identifiants sont liés à un ***widget*** crées dans notre interface ?

**Exercice 2** : modifiez le nom des identifiants des widgets

Par exemple : textView -> messagePourSIO

Attention : propagez les modifications dans l’expression des contraintes de layout.

Etape 4 : manipulation des widgets en Java

Dans notre programme Java, ajoutez 3 variables dans la classe MainActivity, des types suivants :

* Un TextView
* Un EditText
* Un Button

Si la classe n’est pas reconnue, elle s’importe comme sous Eclipse (click sur nom de la class puis Alt+Entrée).

Même règles de visibilité (public …) que pour les programmes Java standards.

**Note** de syntaxe :

Pour récupérer une widget dans un programme java, on utilise la méthode findViewById.

Exemple :

**messageBienvenue** = findViewById(R.id.***sio\_texte***);

Remarque : les versions de sdk antérieures à 26 nécessitent un cast, comme mentionné dans la plupart des tutoriaux. Plus besoin dorénavant : <https://stackoverflow.com/questions/45092494/android-o-casting-to-findviewbyid-not-needed-anymore/45092546>

**Exercice 3** :

Affectez à chaque variable Java la widget correspondante.

A quel endroit se fait l’affectation ? (pas vraiment le choix à ce stade …).

**Partie 3  : action contextuelle sur le bouton**

On souhaite que lorsque l’utilisateur clique sur le bouton, une page s’ouvre avec un contenu dépendant de ce que l’utilisateur a saisi.

Par défaut le bouton n’est pas cliquable tant que l’utilisateur n’a pas rentré du texte dans le champ prévu à cet effet.

Etape 5 : **code à modifier selon les noms que vous avez donné**

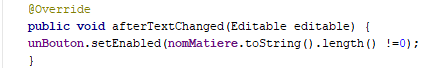
1. Déjà par défaut on désactive le bouton :

**unBouton**.setEnabled(**false**);

1. ensuite on ajoute un Listener d’évènement au champ texte :



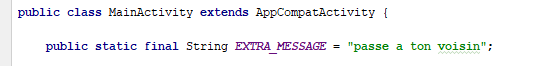
1. A ce stade vous allez voir apparaître 3 méthodes correspondant à 3 types d’évènement différents. On va réactiver le bouton qui était désactivé par défaut.



**Question 2**: à quelle condition le bouton est-il désactivé ?

1. Ajout d’une constante

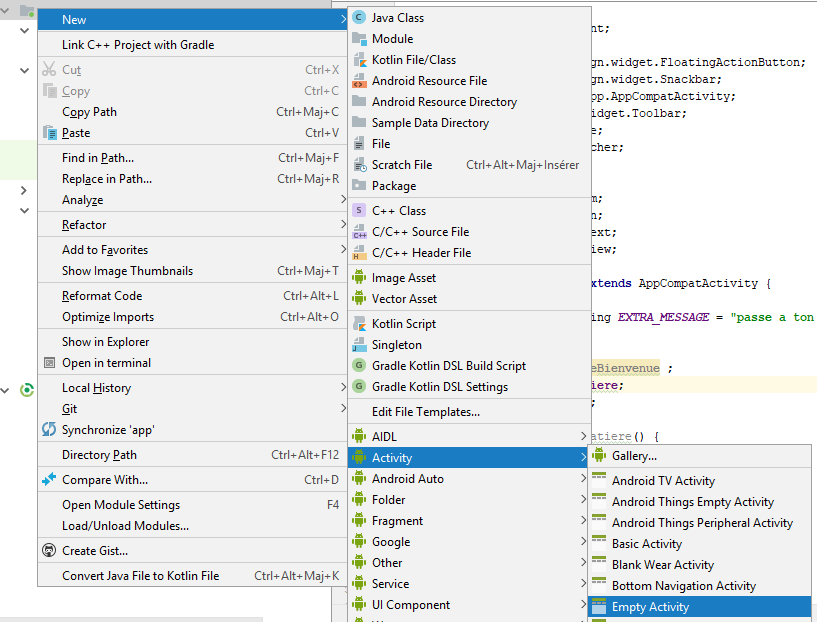
Ajouter après la définition de la classe la constante ( variable statique finale) suivante :



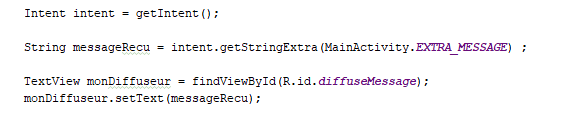
**Question 3**: comment définit on une constante en java ? et en php ? et en Javascript ?

Etape 6 : ajout d’une nouvelle activité qui va réceptionner et afficher le contenu

1. Ajouter une nouvelle activité vide, en démarrant par un clic droit sur **app**, donnez lui un nom (pour moi activity\_message)



1. Dans cette nouvelle activité, ajoutez un TextView, donnez-lui un identifiant, contraignez le
2. Dans le code Java associé à cette activité, insérez le code suivant, dans la méthode **onCreate**() :



Adaptez ce code en fonction de ***l’identifiant*** que vous avez défini à l’étape **b**

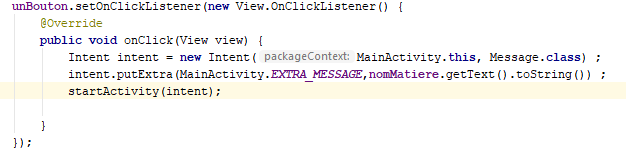
1. Dans le fichier AndroidManifest.xml (situé dans app\manifests), ajoutez l’attribut suivant à la nouvelle activité que vous avez définie



Dans mon cas, j’ai appelé ma nouvelle activité **Message**

Etape 7 : Ajout d’un gestionnaire d’évènement lié au bouton

Dans l’activité principale : **MainActivity.java**



Syntaxe complexe mais à laquelle on s’habitue vite.

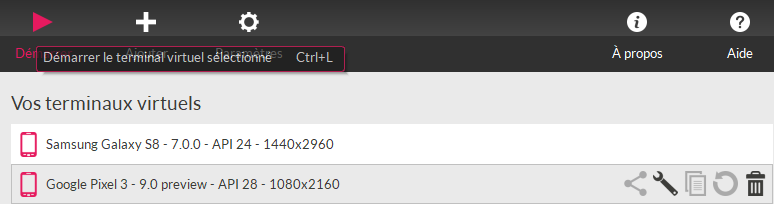
Le 2ème paramètre du constructeur Intent ***doit correspondre*** au nom de la classe de votre nouvelle activité (Message pour moi).

Documentation : ci-joint la documentation officielle pour les listeners : <https://developer.android.com/guide/topics/ui/ui-events>

Les Intents sont un mécanisme permettant de passer des ***messages*** que l’on peut utiliser dans notre application, notamment pour transmettre des messages entre ***activités*** Android.

**Checkpoint n°1 :** validation de l’application

Dans Genymotion ou alors Android Studio, téléchargez le Google Pixel 3 , lancez l’émulateur et testez.



**Partie 4  : utilisation d’un ScrollView**

Evolution de la demande : dorénavant on ne souhaite pas se contenter d’afficher dans la nouvelle activité le texte saisi dans la première, mais on souhaite afficher le contenu d’un document plus conséquent dans un TextView et pouvoir l’afficher en entier.

Etape 8 : modification du layout de la nouvelle activité

Dans la nouvelle activité (chez moi activity\_message), remplacer le ConstraintLayout par un RelativeLayout.

Supprimer également cette ligne (liée au ConstraintLayout)



Supprimer également la contrainte dans le TextView existant.

Etape 9 : affichage d’un document contenu dans une entrée

Rajoutez dans le fichier **strings.xml** une entrée, nommé par exemple *texte\_sio*, et entre les balises insérez le texte du fichier texte associé **slam3.txt**

Remarque : les balises CDATA, maintenant obsolètes, ont été très largement utilisées pendant de longues années et sont encore très présentes.

Ici on en a besoin car la Widget Android TextView n’est pas très adaptée pour afficher du HTML.

**Exercice 4** :

On souhaite maintenant que lorsque on clique sur le bouton : si le text slam3 a été saisi, alors on affiche le document référencé par l’entrée *texte\_sio* du fichier **strings.xml** , et sinon, on affiche le texte saisi.

Remarque : on accède aux strings du fichier strings.xml via R.string.***nomString***

On souhaite un affichage HTML.

**Question 4**: on suppose l’exercice 4 terminé. Quel problème avons-nous ?

Etape 10 : ajout d’un ScrollView

Ajoutez un ScrollView dans le fichier XML, choisissez wrap\_content pour le layout\_width comme pour le layout\_height.

Déplacez la balise fermante du ScrollView de telle sorte que le TextView se retrouve à l’intérieur du ScrollView.

Testez et vérifier que le Scroll se passe bien.

**Exercice 4** :

Complétez le fichier texte slam4.txt en ajoutant du contenu pertinent, sur les éléments suivants : détails des savoir-faire et savoirs associés avec des éléments concrets.

Complétez l’application pour que : lorsque on clique sur le bouton, si le text slam3 a été saisi, le document relatif à SLAM3 soit affiché, si le text slam4 est saisi, le document relatif à SLAM4 soit affiché, sinon affichage du texte saisi.

**Question 5**: explicitation des notions

Composant logiciel :

* cours EMDS prof SIO CNED : <https://www.youtube.com/watch?v=qpQ1tVIQSck> (la vidéo suivante, sur Web Service SOAP, est en revanche obsolète – deprecated)
* fichier PDF : NOTION de COMPOSANT LOGICIEL.pdf. Par exemple Slide 12

Framework :

* Différence Framework et Library : <https://www.geeksforgeeks.org/software-framework-vs-library/>
* Another source : <https://www.youtube.com/watch?v=D_MO9vIRBcA>

1. notion de composant logiciel

Est-ce qu’un composant est équivalent à un objet ?

Qu’est-ce qu’une interface pour un composant ? Est-ce la même chose qu’une interface en programmation OO ?

1. Caractéristiques d’un Framework

Qu’est-ce qu’un Framework ? Quelle est la différence avec un composant logiciel ou une librairie ?

Qu’est-ce que le terme IoC ?

**Partie 5  : ajout d’un radiogroup et de radioboutons**

L’envoi de texte en fonction de la saisie effectuée par l’utilisateur a quelque chose de magique mais ce n’est pas très standard, et le résultat est quelque peu aléatoire.

Nous allons donc ajouter un RadioGroup et 5 RadioButton correspondant aux éléments : slam3, slam4, slam5, ppe et si7

On laissera le bouton de soumission toujours actif, même si aucun texte n’est saisi.

On pensera à committer le code actuel fonctionnel avant de se lancer dans ces ajouts.

Aide si vous n’y arrivez pas, en français :

On créera dans le programme principal un objet RadioGroup et on utilisera la méthode getCheckRadioButtonId pour savoir quel Radio Bouton est cliqué.

On aura besoin d’un LinearView dans le fichier Xml , d’un RadioGroup et de plusieurs radio boutons …