Septembre 2022
Licence Pro PM

TP 1 : Introduction au développement sous Android

Laurent Provot <laurent.provot@uca.fr>

Le but de ce premier TP sera d'appréhender les notions de bases pour créer sous Android Studio un projet simple pour une plateforme Android de type smartphone. Vous vous aiderez pour cela principalement de la documentation que vous trouverez à l'URL suivante : `https://developer.android.com/docs/`

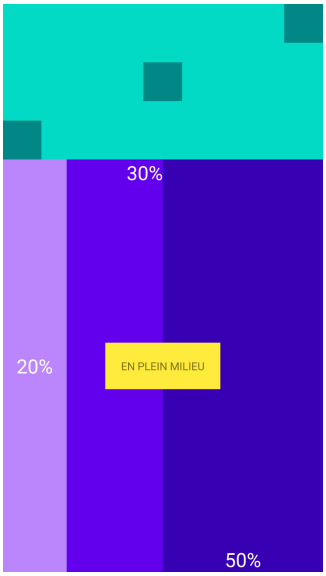
Objectif

L'objectif du TP est de mettre au point une petite application de Quiz à laquelle on répond aux questions par « Vrai » ou par « Faux » (comme sur l'exemple ci-dessous).

L'application étant très simple, le but n'est pas de la terminer le plus rapidement possible, mais au contraire de bien prendre le temps d'explorer les possibilités d'Android Studio, les layouts disponibles sous Android et la documentation relative aux classes utilisées.

Notions abordées : Projet Android Studio, Layouts, Activity, Toast, Button, TextView, resources (strings, drawables), LogCat, Cycle de vie d'une activité

Préliminaires (25-30 minutes max)



1. Créez un projet applicatif Android : vous l'appellerez Quiz, vous le mettrez dans un package `fr.iut.pm` (API minimum 16)
2. Créez une Empty Activity et nommez-là `QuizActivity`
3. Avant de rentrer dans le vif du sujet et de créer la vue du Quiz, commencez par manipuler et comprendre les différents *Layouts* et leurs paramètres. Notamment, vous devrez maîtriser le `FrameLayout`, le `LinearLayout` et le `ConstraintLayout`. De même vous devez connaître les principaux paramètres qui leur sont liés, directement ou indirectement (`orientation`, `gravity`, `layout_width`, `layout_height`, `layout_weight`, etc. et les différentes contraintes). Pour vous aider visuellement, peuplez vos layouts à base de `TextView` et de `FrameLayout` dont vous modifierez l'attribut `background` en lui affectant une couleur, vous aurez ainsi des rectangles de couleurs différentes qui seront bien distinguables. Essayez ainsi de reproduire la vue ci-contre.
   (a) dans une première version n'utilisez pas de `ConstraintLayout`.
   (b) dans une deuxième version (travail à faire à la maison) n'utilisez que le `ConstraintLayout` (1 seul doit suffire).

Mise en place de l'application de base

1. Créez la vue du Quiz en vous inspirant de la capture d'écran de la page précédente
2. Expliquez comment (et où) sont gérés les textes qui apparaissent dans le GUI
3. Ajoutez différents textes pour les boutons Vrai / Faux ainsi qu'une question en dur
4. Explorez le code de la classe `QuizAcivity` pour comprendre commment le layout XML est relié au GUI
5. Ajoutez un comportement à vos boutons pour qu'ils affichent un Toast (consultez la documentation pour plus d'infos, `findViewById` et classe `Toast`). Ce Toast dira si le choix de la réponse est correct ou pas (vous mettrez dans un premier temps cela en dur dans votre code)

Testez votre application en la lançant sur un téléphone ou en utilisant l'émulateur.

Réfactoring

1. Gérer les questions en dur dans le GUI ne respecte pas vraiment les bonnes pratiques vers lesquelles on cherche à tendre lorsqu'on fait de la qualité.
   Créez donc une classe métier `TrueFalseQuestion` pour gérer une question (elle ne devra donc pas contenir de référence à des éléments du framework Android !)
2. Modifiez si nécessaire votre `TextView` pour qu'il possède un ID et soit ainsi manipulable dans le code
3. Ajoutez aussi un nouveau bouton centré sous les 2 précédents pour passer à la question suivante
4. Ajoutez différents intitulés de questions dans le fichier `strings.xml`
5. Créez un *stub* pour simuler une base de données de questions dans lequel vous instancierez un ensemble de `TrueFalseQuestion` en les initialisant avec les questions correspondantes aux intitulés que vous venez de créer.
6. Utilisez ce stub dans votre `QuizActivity` et faites en sorte que votre `TextView` affiche la première question et que le bouton « Suivante » permette de passer aux questions suivantes
7. Ajoutez le code permettant de vérifier que la réponse donnée est correcte. Utiliser un Toast pour le signifier à l'utilisateur

(Avez-vous pensé à refactorer vos sources si un code identique est présent à différents endroits ? Suivez-vous les bonnes pratiques qui vous ont été enseignées ?)

Étude du cycle de vie et persistance

1. Ajoutez une icône (vectorielle) à votre bouton « Suivant » (res/drawable)
2. Lancez votre application, passez 1 ou 2 questions et faites une rotation de l'appareil. Que remarquez-vous ?
3. Ajoutez à `QuizActivity` une constante `TAG` dont la valeur est le nom de la classe
4. Surchargez les 6 méthodes classiques du cycle de vie d'une activité (`onCreate`, `onStart`, `onResume`, `onPause`, `onStop` et `onDestroy`) en loggant le passage dans ces dernières (cf. classe `Log`)
5. Jouez avec le Home button, le Back button et faites des rotations de l'appareil, observez, déduisez-en une explication à votre observation du point 2.
6. Profitez-en pour manipuler un peu LogCat et testez ses fonctionnalités (notamment le filtrage).
7. Ajoutez un layout différent pour le mode paysage (res/layout-land) en faisant en sorte que le bouton « Suivant » soit collé en bas à droite.
8. Vérifiez que celui-ci est utilisé lors du passage en mode paysage.
9. Gérez la sauvegarde des données pour qu'après une rotation de l'appareil le quiz ne reparte pas de la première question.
   Indice : `onSaveInstanceState(Bundle outState)` et `onCreate(Bundle savedInstanceState)`