L3 - Programmation Android

# TP n°4

## Application de ventes immobilières

## Présentation

L'application qui va être développée au cours des TPs à venir est une application de ventes immobilières. Les fonctionnalités de base pour ce TP sont les suivantes :

- Écran d'accueil avec boutons pour lancer les divers écrans de l'application (à faire évoluer plus tard)
- Visualisation d'un bien immobilier à vendre (pris au hasard pour commencer)
- Saisie d'un bien immobilier à vendre

On ajoutera ensuite :

- Liste des bien à vendre avec, pour chaque annonce, la possibilité de voir le détail et de contacter l'agent immobilier (email/tél.)
- Avoir un profil local pour ne pas saisir son email/tél. à chaque dépôt d'annonce
- Sauvegarder localement des annonces sur l'appareil
- Prendre une photo avec l'appareil pour illustrer une annonce lors du dépôt
- Recherche de biens à vendre
- Etc.

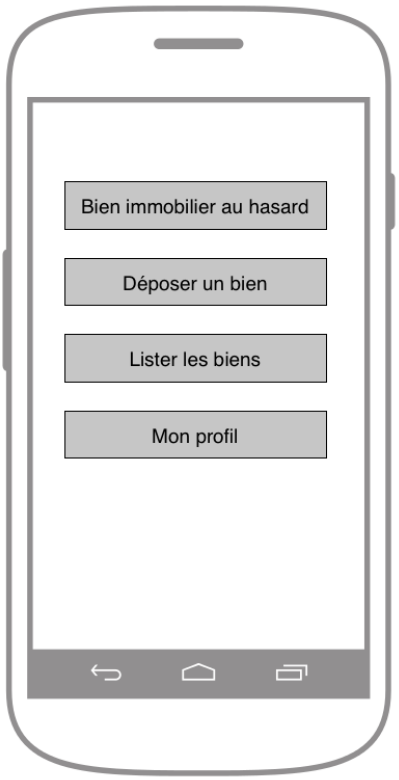
Les TPs à venir préciseront les fonctionnalités complémentaires.

## Constitution des groupes

Les groupes de travail doivent être constitués de **3 ou 4 étudiants**. Il ne sera pas autorisé d'avoir des groupes de plus de 4 personnes.

Une fois votre groupe constitué, envoyez un mail à Jean-Marc Lecarpentier et à votre chargé de TP avec la liste du groupe (indiquer NOM Prénom n° étudiant). Pour rappel les mails des intervenants : lecarpentier@unicaen.fr, youssef.chahir@unicaen.fr et tanguy.godquin@ensicaen.fr

# Démarrage




Créer un projet Android et une Activity qui, pour commencer, servira à lancer diverses parties de l'application et sera modifiée ultérieurement. On pourra commencer avec 4 boutons comme sur le schéma ci-contre.

# Modèle d'une annonce

Créer une classe Java `Propriete` qui modélise une annonce.

Pour commencer, une annonce sera composée de :

- id : l'identifiant de l'annonce (une chaine de caractères)
- titre
- description
- nombre de pièces principales
- liste de caractéristiques (sous forme de chaînes de caractères, par exemple "garage", "piscine", etc)
- prix : un entier
- ville
- code postal de la ville
- vendeur : un objet modélisant un vendeur (voir ci-dessous)
- images : une liste d'URLs d'images illustrant l'annonce. La liste peut être vide.
- date : la date de dépôt de l'annonce

Créer une classe Java `Vendeur` qui modélise un vendeur avec :

- id (une chaine de caractères)
- nom
- prenom
- email
- telephone

# Visualisation d'une annonce

### 1- Visualisation d'une annonce


Créer une Activity pour visualiser une annonce, par exemple comme sur le schéma ci-contre. Cette Activity est lancée par un clic sur le bouton « Bien immobilier au hasard ».

Vous êtes libre de disposer les éléments de l'annonce comme vous le voulez. Penser à rendre la vue « scrollable » puisque le contenu ne tiendra pas dans un seul écran.

Pour commencer, créer dans l'activité une instance de `Propriete` fictive qui servira pour remplir la vue, avec au moins une image. Pour

l'affichage de l'image, intégrer une bibliothèque (Glide, Picasso ou Fresco) et afficher la première image de la liste pour commencer.

Gérer ensuite le cas où l'instance de `Propriete` n'a aucune image. On pourra alors soit afficher une icône image par défaut ou bien ne pas avoir de zone d'image.

### 2- Requête HTTP et récupération d'un bien immobilier

L'API de gestion des biens immobiliers sera mise en ligne ultérieurement.

Pour commencer, une « mock API » sera utilisée. Il s'agit d'un fichier JSON qui contient des données au format qui sera utilisé par l'API.

La réponse de l'API au format JSON est constituée de 2 propriétés :

- success : valeur true/false
- response : les données transmises. Contient le message d'erreur si success est false.

On pourra utiliser les URLs suivantes pour le développement en cours.

Réponse de type erreur :
https://ensweb.users.info.unicaen.fr/android-estate/mock-api/erreur.json

Réponse avec les données d'une annonce :
https://ensweb.users.info.unicaen.fr/android-estate/mock-api/immobilier.json

Intégrer la bibliothèque OkHttp au projet et interroger la mock API pour obtenir les informations sur une annonce au format JSON.

Intégrer la bibliothèque Moshi au projet et l'utiliser pour créer une instance de `Propriete` à partir du JSON reçu et utiliser cette instance pour l'affichage.

## Pour les plus avancés

Google a lancé Jetpack en mai 2018. Jetpack (https://developer.android.com/jetpack/) est un ensemble de composants pour accélérer le développement Android.

Le composant Data Binding permet de remplir les vues Android directement à partir des instances des classes Java qui modélisent les données. Intégrer ce composant pour l'affichage d'un bien immobilier à partir d'une instance de `Propriete`.

Voir https://developer.android.com/topic/libraries/data-binding/ pour démarrer.