Travail pratique #3
“Car Retail”

Objectifs

À la fin de ce travail pratique, l’étudiant devrait:
• Être en mesure de gérer une liste
• Être en mesure d’effectuer une requête GET sur un serveur
• Être en mesure d’effectuer une requête POST sur un serveur
• Être en mesure d’intégrer l’affichage d’image dans une interface
• Utiliser des libraires externes dans un projet

Introduction

Le but de ce troisième travail pratique dans le cours d’Informatique Mobile et Application (IMA) est connecter votre application à internet. Vous devriez par la suite être en mesure de faire des requêtes à des services web et de consommer des données.

Énoncé

1. Créez un nouveau projet Android à partir d’un “Empty Activity”. Ensuite cliquez bouton de droite dans la section de code de gauche. Effectuez New -> Activity -> Tabbed Activity. À l’étape qui offre de personnaliser l’activity faites bien attention de sélectionner “**Action Bar Tabs (with ViewPager)**” au niveau du style de navigation sinon vous aurez plus de difficulté à intégrer le TabBar.
2. Dans le TabBar vous allez avoir 3 “tabs”. Vous pouvez voir une représentation graphique de ce que vous devez faire plus bas:
    1. **Offres**: permets de consulter toutes les offres sous forme de liste(Liste Marques->Liste Modèle->Liste Offres->Détail de l’offre)
    2. **Vendre**: permets de remplir un formulaire pour ajouter une nouvelle offre
    3. **Mes Annonces**: Affiche une liste de vos offres présentement sur le serveur.(Liste Offres->Détail de l’offre)
3. Assurez-vous de retirer tout artifice de navigation qui sont inutile et qu’Android studio a créé pour vous. Par exemple le bouton flottant “courriel”.
4. Modifiez les couleurs de base de l’application pour avoir un mélange de rouge et blanc dans l’Action Bar. Voir aperçu plus bas. Voici quelques teintes de rouge qui s’agencent bien ensemble: #e20000, #7d0000,#b9120d.
5. Dans l’onglet Offres, il y aura quatre (4) pages qui permettront de consulter toutes les offres:

    1. Une liste des marques de voitures
    2. Une liste des modèles pour une marque
    3. Une liste d'offres en fonction du modèle sélectionné
        1. Votre cellule ici devrait afficher un minimum d'information. Il est de votre responsabilité de décider le contenu à y afficher.
    4. Une page de description de l'offre. Vous devez afficher l'image qui est associée à l'offre et sa **fiche complète** (description, vendeur, etc) qui est uniquement disponible sur la méthode de détail dans l'API.
        1. Un bouton contact doit également être présent et ouvrir une application de courriel avec l'adresse courriel du vendeur.
6. Dans l'onglet "Vendre" il y aura une (1) seule page:
    1. Une page de formulaire pour permettre de rajouter des offres.
    2. Pour des fins de correction, chacun de vos champs doit avoir une valeur par défaut à votre choix.
    3. Chaque champ doit être validé dans l'interface avec des valeurs logiques.
7. Le placement de l'information et l'information affichée dans les pages et les cellules est libre et à votre choix. Vous devez être logique et respecter les concepts que nous avons vus en classe. Par exemple, vous ne devriez pas tout placer dans un même "pain" et vous devriez plutôt espacer vos informations pour faciliter la lecture. N'hésitez pas à vous inspirer des applications existantes pour vous aider et/ou du TP2
8. Dans l'onglet "Mes annonces", il y aura deux (2) pages:
    1. Une liste de vos offres qui reprend le même affichage que lors de la consultation des offres "générales".
    2. Une page de description de l'offre qui reprend le même affichage que lors de la consultation des offres "générales".
9. Certaines méthodes sont protégées par un header d'authentification. Pour vous authentifier vous devez appeler la méthode `/account/login/` avec vos informations personnelles de l'Université Laval. Votre courriel est le courriel qui vous a été fourni par l'Université Laval et le numéro d'identification est celui qui est sur votre carte étudiante(composé de 9 chiffres).
    1. Le token qui sera retourné par la méthode login doit être inclus dans le header Authorization avec la valeur "Basic MON_TOKEN" comme discutée en classe.
    2. Le token expire après un certain temps(maximum 24h) vous ne pouvez donc pas le hardcoder dans votre application.
    3. Lors d'un "challenge" d'authentification(le serveur vous retourne alors un HTTP 401) vous devrez afficher un AlertDialog avec 2 champs EditText qui contiendront le courriel et le numéro d'identification de même qu'un

bouton Connexion. Le bouton Connexion effectuera l'appel au login. Une fois le login effectué votre application doit poursuivre d'elle même l'action que vous alliez effectuer.
    1. Les champs dans l'AlertDialog tout comme le reste de l'application doivent être renseignés par défaut pour fin de correction.
    2. Un aperçu de ce que pourrait ressembler cet AlertDialog est disponible plus bas.

10. Une librairie comme Picasso pourra être d'une grande aide pour le chargement d'image, mais vous pouvez utiliser une autre librairie. Même chose au niveau d'une librairie réseau vous pouvez utiliser OkHttp.
11. La présentation est à votre convenance, mais vous devez être logique. Par exemple un formulaire doit contenir des validations et un utilisateur ne peux pas deviner qu'il faut inscrire AT pour automatique dans le choix de transmission.
12. Vous pouvez utiliser des ListView ou des RecyclerView, mais vous devez absolument bien intégrer le concept de **ViewHolder**.
13. Vous n'avez aucune restriction au niveau de l'utilisation des Fragments et/ou Activity, à l'exception du ViewPager dont les onglets sont obligatoirement des Fragments. Vous pouvez également les mélanger. Vous devez par contre vous assurer que le bouton back du téléphone réagisse correctement.
14. Assurez-vous que tout le contenu de votre application sera visible sur tout type de téléphone, petits et gros!
15. Soyez lâche et réutilisez vos classes!

Consigne supplémentaire:

- Votre code source doit être **obligatoirement** fourni dans un bundle de type git. Vous n'êtes pas obligé d'utiliser git à son plein potentiel (soit d'avoir plusieurs commit), mais vous devez au moins fournir votre code dans ce format.
- Vous devez signer votre APK avec le keystore créé dans le TP1; si par mégarde vous l'avez perdu vous pouvez en généré un nouveau en suivant les requis imposé dans le TP1.
- Votre code doit être propre et vos choix de nom de classes/ variables doivent être en contexte avec le travail demandé.
- Prenez note que les commentaires tout au long du code, bien que toujours une bonne pratique ne sont pas obligatoires.
- Un APK retourné avec un keystore de type Debug sera refusé et considéré comme invalide.
- Votre nom de package **doit** être ca.ulaval.ima.tp3
- Un guide de procédure de remise est disponible sur le portail des cours dans la section "Contenu et activités" -> Guides. Veuillez vous y référer pour vous assurer d'une remise conforme.
- Votre code source doit être écrit en JAVA uniquement avec le SDK Android et compatible avec l'API 16.

- Si votre APK de remise ne fonctionne pas ou bien qu'il est signé avec une clé de Debug, la note de 0 vous sera assignée pour la section "Réalisation de l'application".
- Le travail doit être réalisé individuellement
- Votre bundle git doit se "unbundler" avec la commande suivante "git clone nom_du_bundle" sinon vous recevez la note de 0 pour toute évaluation de code. La même chose s'appliquera si vous ne remettez pas de bundle.

Remise

Ce que vous devez remettre à la fin du travail pratique
- Un fichier zip qui contient votre APK qui est signé avec votre propre keystore et le bundle git qui contient votre code source(sous la forme d'un projet Android Studio)

La nomenclature de votre dossier de remise **doit obligatoirement** être de cette forme:
- Fichier zip à votre nom
  - L'application en format APK nommé "TP3.apk"
  - Le bundle git nommé"TP3.bundle"

La remise de votre travail doit être effectuée sur le portail des cours. (https://www.portaildescours.ulaval.ca) Une boite de dépôt est disponible dans la section "Évaluations et résultats".

Barème de correction

15% de la note finale est allouée pour ce travail pratique

	Note
Qualité du code source (Nom des objets, respect des normes de programmations JAVA, réutilisation de code)	/3
Concept de programmation - Liste intégré avec ViewHolder	/2 /2
Réalisation de l'application testé via APK - Présentation des offres(Marques, Modèles, liste d'offres, ViewPager) - Page de description d'offre - Présentation page de formulaire - POST fonctionnel - Login/Authentification - Validation de champ	/22 /6 /4 /4 /3 /3 /2
Respect des consignes (signature de l'APK, format de remise)	/3
Total	/30

API

- Documentation disponible sur cette URL: http://159.203.33.206/docs
- URL de base de l'API : http://159.203.33.206/api/v1/

Aperçu

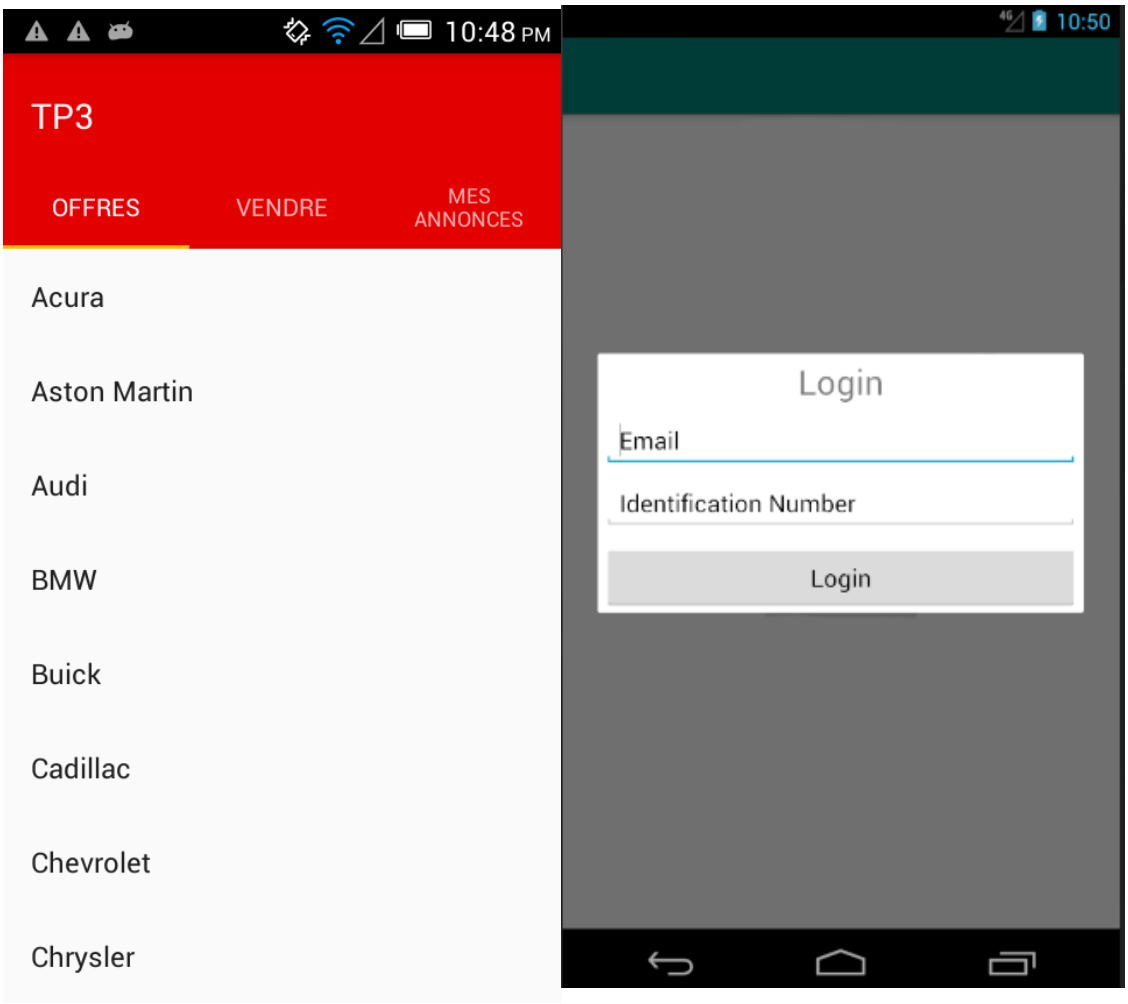
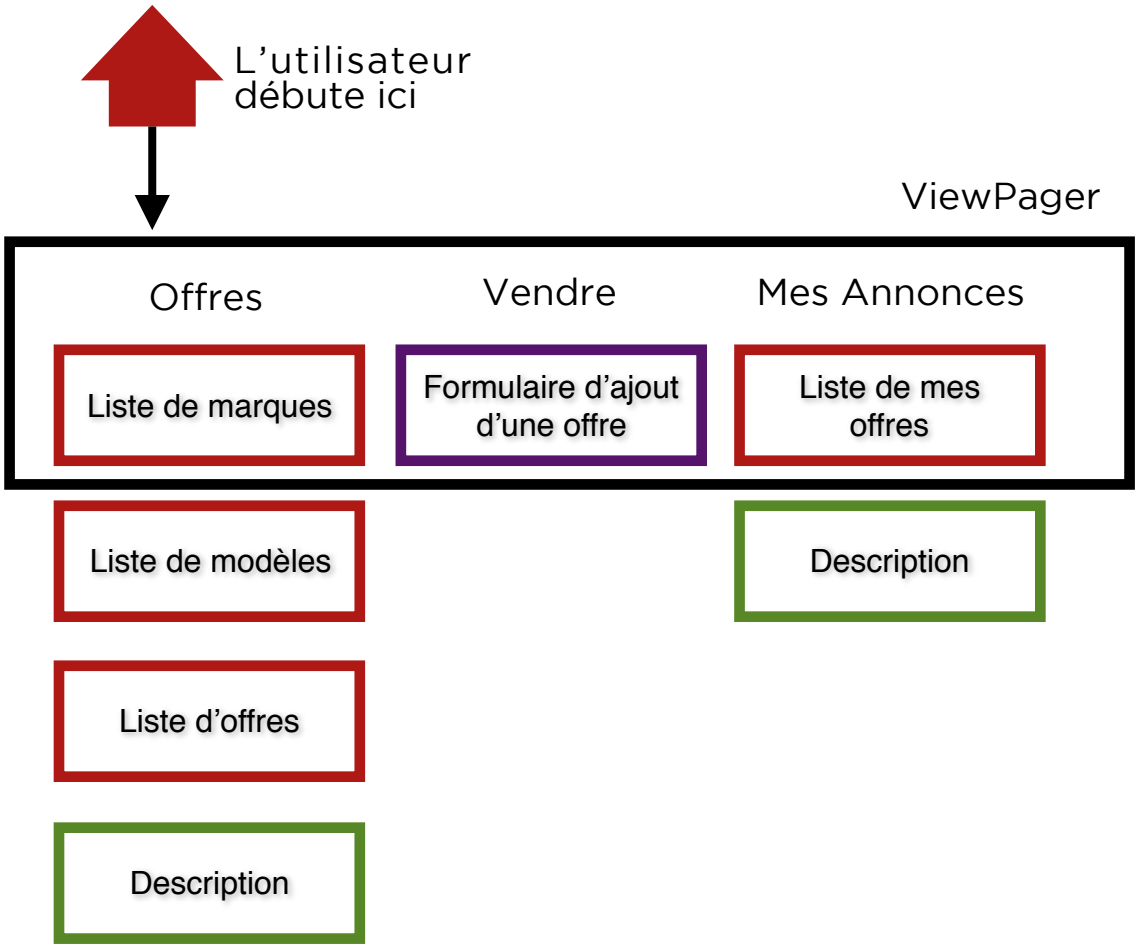


Schéma de l'application



Si on effectue un back dans le ViewPager on quitte l'application. Sur le reste des pages on retourne au parent.