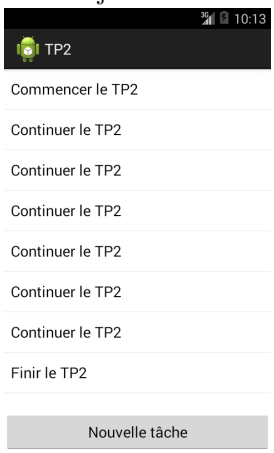
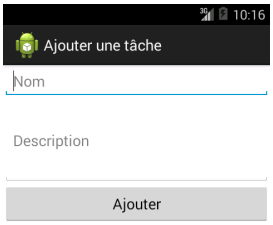
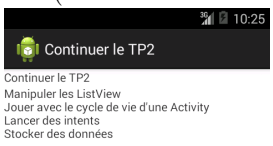
Android - TP2

Source, pdf et corrigé de ce TP :
http://tiny.cc/techmob

L'objectif de ce TP est de réaliser une todo list (liste de choses à faire).







Ce TP fera appel aux notions suivantes :

- ListView
- Listeners
- Cycle de vie de l'activity
- Intents
- Stockage de données

Il est fortement conseillé d'avoir le cours à portée de main et à ne pas hésiter à se référer à la documentation officielle http://d.android.com ainsi qu'à la multitude de tutoriaux disponibles.

1 L'écran principal : affichage des tâches

- Créer un nouveau projet android avec une Activity simple (EmptyActivity)
- Modifier le layout de cette activity pour y placer une ListView qui prendra toute la largeur et toute la hauteur

1. Quel composant est chargé de faire le lien entre les données et la ListView ? (on l'implémentera plus tard)

On souhaite ajouter un bouton "ajouter une tâche" en dessous de la ListView.

2. Pourquoi un LinearLayout "naïf" ne convient t'il pas ?
3. Pourquoi un RelativeLayout ne convient t'il pas totalement ?

- En utilisant la propriété layout_weight de LinearLayout, placer le bouton en bas de l'écran.
- Ecouter les clicks sur le bouton et afficher un Toast à chaque click

2 Un écran secondaire : ajout d'une tâche

- Créer une deuxième activity (rappel : ne pas oublier de l'ajouter au manifest !)
- Ajouter deux champs de texte modifiables (nom et commentaires) ainsi qu'un bouton "ajouter" à cette activity (libre à vous de définir le layout)

3 Enchainement des écrans

3.1 Dans un sens ...

5. On veut lancer l'activity d'ajout d'une tâche lors d'un click sur le bouton "ajouter une tache" de l'activity principale. Quel concept android va-t-on utiliser ? (indice : il peut être soit implicite soit explicite)
6. Dans ce cas, est-il implicite ou explicite ?
7. Pensez vous qu'il faille passer des données lors de l'appel à l'activity "ajouter une tâche" ? Si oui, lesquelles ?

- Lancer l'activity d'ajout de tâche lors d'un click sur le bouton.

3.2 ...et dans l'autre

Lors d'un click sur le bouton "ajouter" de l'activity d'ajout, on souhaite retourner à l'écran principal.

8. Est-il judicieux de lancer un intent vers l'activity principale ?

En réalité, android construit une pile des activités lancées (stack). Documentation officielle sur le stack

- En faire l'éxpérience en allant sur l'activity d'ajout puis en appuyant sur le bouton retour de l'appareil.
- Utiliser la méthode finish() de Activity pour fermer l'activity et ainsi revenir à l'écran principal lors d'un click sur le bouton ajouter.

4 Place aux données !

- Créer une classe métier déstinée à contenir les données : Element. Chaque élément de la todo list contiendra au moins un nom et une description.

4.1 Afficher les données dans la ListView

On ne souhaite afficher dans la liste que le nom des éléments.

9. Au vu de la simplicité des données à afficher pour chaque élément (un seul champ de texte), quel type d'adapter vous parait le plus intéressant ? (Rappel : on a le choix entre ArrayAdapter, CursorAdapter et faire sa propre implémentation à partir de BaseAdapter)

- Dans onCreate : Créer une instance de l'adapter correspondant en utilisant comme layout pour chaque view le layout "android.R.layout.simple_list_item_1"
- Ouvrir le fichier XML de ce layout à l'aide de CTRL+click et constater sa simplicité
- Initialiser cet adapter avec une liste de Element générés à la volée
- Affecter cet adapter à la ListView

4.2 Stocker les données

10. Quelle solution de stockage des données vous parait la plus adaptée ? (Rappel : on a le choix entre Preferences, fichier plat, base de données SQLite et stockage distant)

- Pour des raisons de simplicité, on va opter dans la suite pour un stockage en fichier plat (libre à vous de le remplacer par une base SQLite)
- Créer une nouvelle classe ElementDAO qui se chargera des sauvegardes / chargements de données.
- Y créer une méthode qui écrit une liste d'éléments Element dans un fichier en utilisant un format de stockage primitif par exemple nom#description et un Element par ligne.
  Note : Ce format est un très mauvais choix en pratique car peu robuste et peu extensible mais suffira largement ici.
- Créer une méthode qui instancie une liste d'Element à partir des valeurs lues dans le fichier.
- Dans la méthode onCreate(), remplir la ListView à partir des données contenues dans le fichier.

4.3 Remplir les données

- Lors d'un click sur le bouton "ajouter" de l'activity d'ajout, créer une instance d'Element et la rajouter au fichier. (pour simplifier, réécrire la totalité du fichier à chaque fois)

11. Pourquoi la ListView n'est elle pas mise à jour lors du retour sur l'activity principale ?

- Utiliser le cycle de vie des activities pour recharger les données de la ListView lors du retour sur l'activity principale. (implémenter la bonne méthode onXXXXX)

5 Voir le détail d'un élément

- Créer une activity de visualisation d'item avec les champs utiles (nom, description ...)
- Récupérer les clicks sur les éléments de la ListView (Attention à bien utiliser onItemClickListener et non onClickListener).
- Lancer l'activity de visualisation lors d'un click sur un item de la liste
- Transmettre à l'activity le nom de l'item choisi
- Récupérer les données dans l'activity de visualisation et afficher les données de l'élément

6 Pour aller plus loin

Wow, déjà tout fait ? Voici quelques pistes d'améliorations :

- Le stockage en fichier plat n'est pas très judicieux. Mettre en place une base de données SQLite à la place.
- Les données associées aux items sont pour l'instant minimales, pourquoi ne pas ajouter un identifiant, une date d'ajout, un statut (en cours, fait ...), une priorité, une catégorie ... ?
- L'affichage des éléments dans la liste est basique mais on peut difficilement faire mieux avec ArrayAdapter. Créer un adapter maison en héritant de BaseAdapter
- Partager les items par mail, sms, twitter ...en ajoutant un intent implicite dans l'activity de visualisation ou de modification.