Université d'Oran 1
Faculté des Sciences Exactes et Appliquées
Département d'Informatique

3ème année Licence – Semestre 2 **Année : 2019/2020**
Module : Applications Mobiles

Fiche TD/TP n°5 : Création d'interfaces avancées

Objectif :

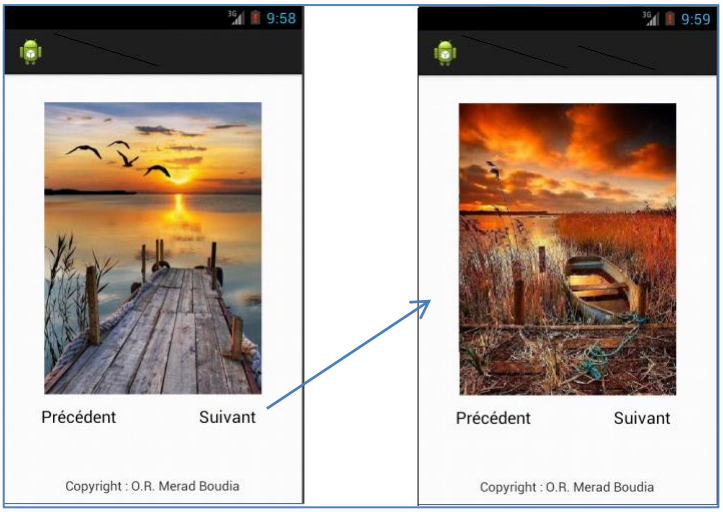
- Le but de ce TD/TP est de se familiariser avec les éléments les plus importants dans la création d'interfaces graphiques.

Prérequis : Du cours 5 au cours 9.

Exercice 1 :

- En suivant les étapes de création d'un **ImageSwitcher** (Cours 9). Créez un projet Android qui permet d'afficher des images en utilisant l'animation suivante (l'image glisse de gauche à droite lorsqu'une autre image est sélectionnée) :

```
android.R.anim.slide_in_left
android.R.anim.slide_out_right
```



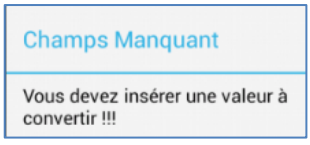
Exercice 2 :

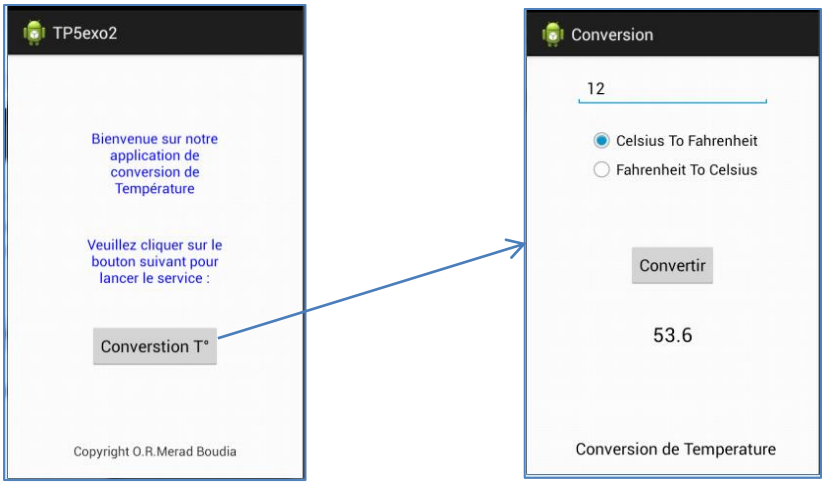
- Créez un nouveau projet Android qui permet de convertir la température entre Celsius et Fahrenheit.
- Les Formules utilisées pour la conversion de Celsius à Fahrenheit et inversement:

$$Tc = (5/9)*(Tf-32)$$

$$Tf = (9/5)*Tc+32$$

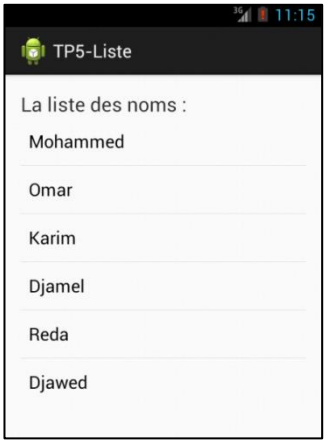
- Ajoutez une AlertDialog (Voir Cours 9 – les popups) pour l'erreur « champ vide ».
- Ne pas oublier le copyright.
- Exemple d'exécution :





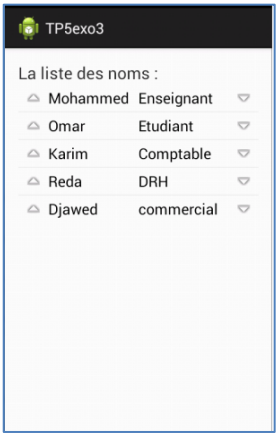
Exercice 3 :

- Créez un projet Android qui permet d'afficher la liste des noms présentée dans la figure suivante :



I. Utiliser un adapter personnalisé

On peut utiliser le layout standard **simple_list_item_2** qui contient quant à lui deux TextView positionnés l'un sous l'autre et ayant pour id **text1** et **text2**. Mais il est plus intéressant de créer et utiliser son propre **layout**. L'objectif est d'obtenir la liste présentée dans la figure suivante :



1. Ajout d'une classe métier Personne

- Ajoutez à votre projet une classe **Personne**.
- Définissez deux attributs privés de type String : **nom** et **fonction** et Générez le constructeur et **les getters**. (utilisez l'outil Eclipse : Clic droit > Source > Generate…)

2. Ajout d'un Layout personnalisé pour afficher chaque personne

- Ajoutez un nouveau fichier de ressources dans le dossier **res/layout** de votre projet ; nommez ce fichier **detail_personne.xml**.
- Dans ce layout, insérez un LinearLayout avec une orientation horizontale et ajoutez à l'intérieur les éléments graphiques suivants :

ImageButton avec id btnHaut et poids=1
TextView avec id tvNom et poids = 3
TextView avec id tvFonction et poids = 3
ImageButton avec id btnBas et poids = 1

- Pour la propriété background des boutons vous utiliserez les ressources Drawable système existantes arrow_up_float et arrow_down_float
- Modifiez les propriétés du LinearLayout: paddingTop et paddingBottom à 4dp.

3. Création d'un Adapter personnalisé

- Ajoutez une nouvelle classe **MonAdapter** à votre projet.
- Cette classe hérite de la classe **ArrayAdapter<Personne>**.
- Générez le constructeur de la classe; vous utiliserez l'avant dernier de la liste.
- Déclarez les trois attributs privés suivants dans votre classe :

Context context ;
int idLayout ;
List<Personne> liste ;

- Initialisez ces attributs dans le constructeur que vous venez de créer.
- Implémentez la méthode getView( ) (clic droit > Generate > Override Methods …
- Implémentez le code de la méthode getView() (voir cours)

Indication :

```
Personne P = liste.get(position);
TextView nom = (TextView) v.findViewById(R.id.tvNom);
nom.setText(P.getNom());
TextView fonction = (TextView) v.findViewById(R.id.tvFonction);
fonction.setText(P.getFonction());
```

4. Créer l'adapter et l'associer à la ListView

- Ouvrez la classe MainActivity et remplacer le code précédent : initialisez les données, créez l'Adapter et liez le à la ListView (Voir cours).

II. Gérer le clic des boutons de chaque élément de la ListView

Nous allons maintenant nous intéresser à nos boutons haut et bas. Ils doivent nous permettre de déplacer les éléments de la ListView vers le haut ou vers le bas. Autrement dit, nous allons devoir effectuer des mises à jour dans notre liste de personnes (supprimer un élément et le recréer à un autre index) puis mettre à jour la ListView. C'est donc ici que la méthode **notifyDataSetChanged** va intervenir.

- Dans la classe MonAdpater, Ajouter à la méthode getView() les instructions suivantes :

```
ImageButton haut = (ImageButton) v.findViewById(R.id.btnHaut);
haut.setTag(position);
ImageButton bas = (ImageButton) v.findViewById(R.id.btnBas);
bas.setTag(position);
```

- Définir les propriétés **onClic** pour les deux éléments ImageButton : méthode clicHaut pour le bouton btnHaut et méthode clicBas pour le bouton btnBas.

- Ajoutez les méthodes correspondantes dans la classe MainActivity. Testez et vérifiez que vous récupérez bien l'index de l'élément cliqué :

```
public void ClicBas (View v){
//Traitement
Toast.makeText(MainActivity.this, " Vous avez cliqué sur l'élément "+
v.getTag(),Toast.LENGTH_SHORT).show();
}
public void ClicHaut (View v){
//Traitement
Toast.makeText(MainActivity.this, " Vous avez cliqué sur l'élément "+
v.getTag(),Toast.LENGTH_SHORT).show();
}
```

-Il n'y a donc plus qu'à modifier la liste de personnes pour déplacer l'élément sélectionné vers le début ou vers la fin puis à mettre à jour la ListView.

- A vous … Vous disposez des méthodes **remove** et **add** sur les objets List<>. N'oubliez pas de mettre à jour la ListView en appelant la méthode **notifyDataSetChanged** de l'adapter.

✓ **Indications** :

```
int a = (Integer) v.getTag();
liste.remove(a);
adapter.notifyDataSetChanged();
```