TP N° 4

1. ListView

ListView est un composant android qui permet d'afficher les listes d'informations. L'utilisateur peut sélectionner les items de la liste par simple clic.

1.1 Adapter

La liste des items est insérée automatiquement dans le ListView en utilisant l'adapter qui insère les informations à partir d'une source comme tableau ou base de donnée.

1.2 Attributes de ListView

Voici quelques attributs les plus utilisées pour assurer la bonne représentation des ListViews.

```
<ListView
android : id = "@+id/simpleListView"
android : layout_width = "fill_parent"
android : layout_height = "wrap_content"
android : divider = "#00ffff"
android : dividerHeight = "1dp"
android : listSelector = "#00ffff" />
```

divider: séparateur entre les item, dividerHeight :l'épaisseur d'un séparateur, listSelector: la couleur attribuée à l'item séléctionné.

1.1.2 ArrayAdapter

ArrayAdapter est l'adapter adéquat pour les listes simples dont les éléments sont des chaines de caractères, il dispose par défaut un layout contient un seul textview. ArrayAdapter est créer comme suite :

ArrayAdapter adapter = new ArrayAdapter < String >(this ,android.R.layout.simple_list_item_1, String Array);

Si vous voulez utilisé des vues plus complexe on doit utiliser des adapters personnalisés (Base adapter).

1.3.Base Adapter

Les listes personnalisées nécessitent la création de ses propres adapter en héritant Base Adapter.

1.3.1 Création de Base adapter :

L'utilisation des listes personnaliser doit passer par l'implémentation d'un adapter personnalisé suivant les étapes :

Etape 1 : création de la classe des élément de la liste avec le layout qui contient les éléments de l'interface correspond à chaque propriété.

Etape 2 : Création de l'adapter personnaliser en héritant Base adapter dans lequel on doit fournir une implémentation des deux méthodes getCount () : pour compter le nombre des éléments de la liste et getView() pour la création des éléments de l'interface correspond au objet de la liste.

Etape 3 : attribuer l'adapter au listview comme suit :

```
ListView lv=(ListView) findViewById(R.id.listep);
ContactAdapter myAdapter;
myAdapter=new ContactAdapter(this, R.layout.contact_lay, mylist);
// Création de l'adapter personnalisé
public class ContactAdapter extends ArrayAdapter<Contact> {
    ArrayList<Contact> contactlist;
    LayoutInflater inflat;
    int Rc;
    public ContactAdapter(Context c, int ressource, ArrayList<Contact> L)
{
        super(c, ressource,L);
        inflat=( LayoutInflater)
c.getSystemService(c.LAYOUT_INFLATER_SERVICE);
        Rc=ressource;
        contactlist=L;
    }
public View getView(int position, View convertView, ViewGroup parent){
        View v=convertView;
        if (v==null) v=inflat.inflate(Rc,null);
    TextView tvName=(TextView)v.findViewById(R.id.TV_Nom);
    TextView tvPrenom=(TextView)v.findViewById(R.id.TV_Prenom);
        tvName.setText(contactlist.get(position).getNom());
    tvPrenom.setText(contactlist.get(position).getPrenom());
        return v;
}
public int getCount(){return contactlist.size() ;}

}
```

1.4 L'ajout de l'évènement onclicklistener au listview

Pour faire un traitement en cliquant sur un item d'un listeview lv nous utilisons la méthode suivante.

```
listeview.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    public void onItemClick(AdapterView<?> parent, View view,
                            int position, long id) {
        // traitement à faire.
    }
});
```

2. Travail demandé

Créer une application qui permet d'afficher la liste des contacte disponibles sur la machine. Chaque entré de la liste contient principalement le nom et le prénom d'un contact. En cliquant sur un contact, des informations supplémentaires comme la photo et le n° de téléphone s'affiche dans une deuxième activité. L'application offre la possibilité de modifier ces dernières informations avec des interfaces convenables.