

# Département d'Informatique, L3, S6

## Cours développement mobile

### Afficher les listes de données avec **ListView**

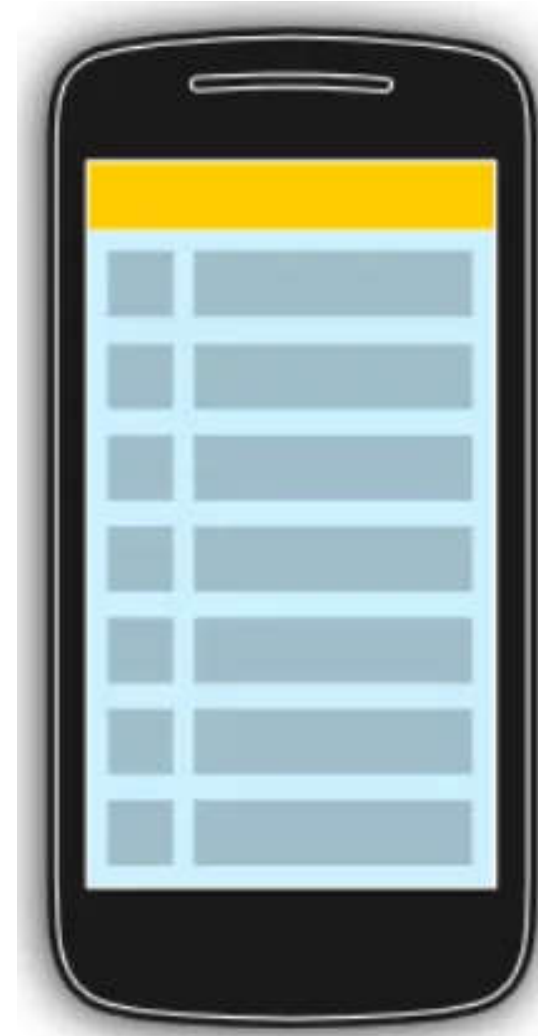


# Besoin

- Un ensemble de données de même type stockées sous forme d'une liste.
  - Liste d'employés
  - Liste de médicaments,
  - Liste de clients
  - etc.

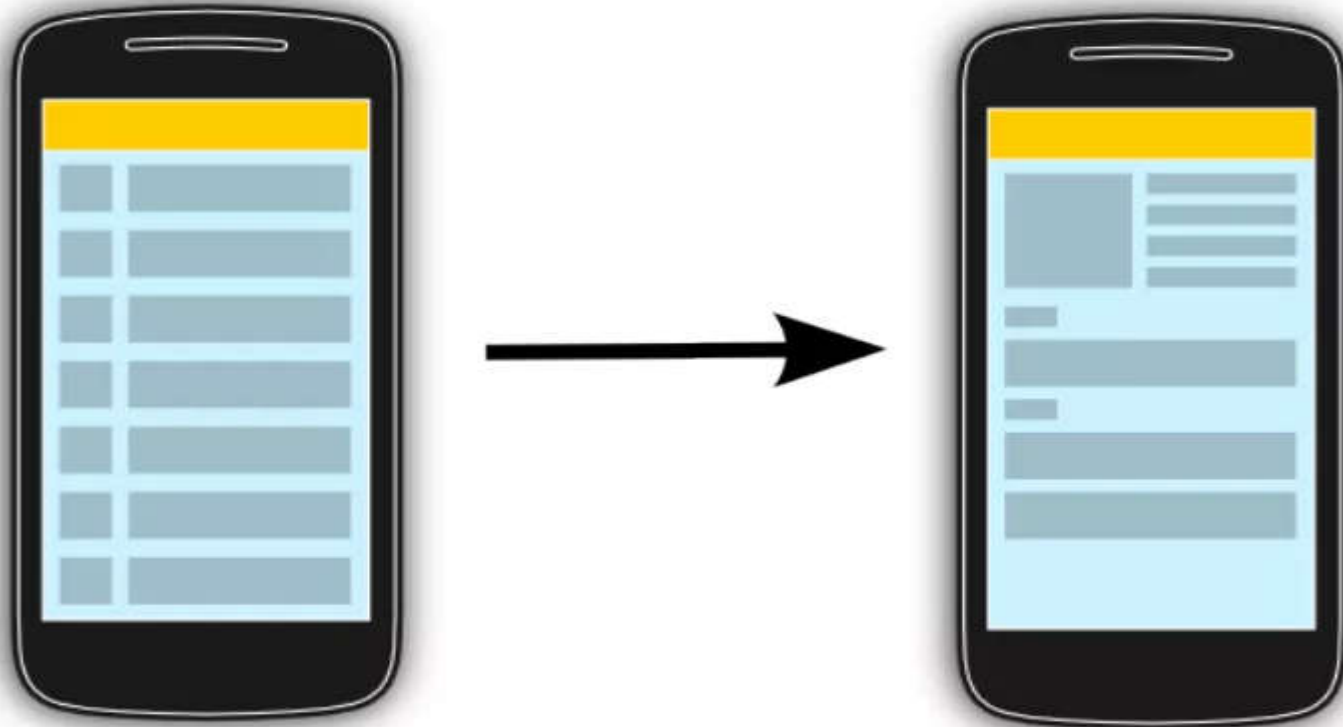
# Quoi faire

- Pour pouvoir afficher des interfaces de ce type:



# Des listes dynamiques et interactives

- Liste d'éléments-- > le clique nous affiche les détails

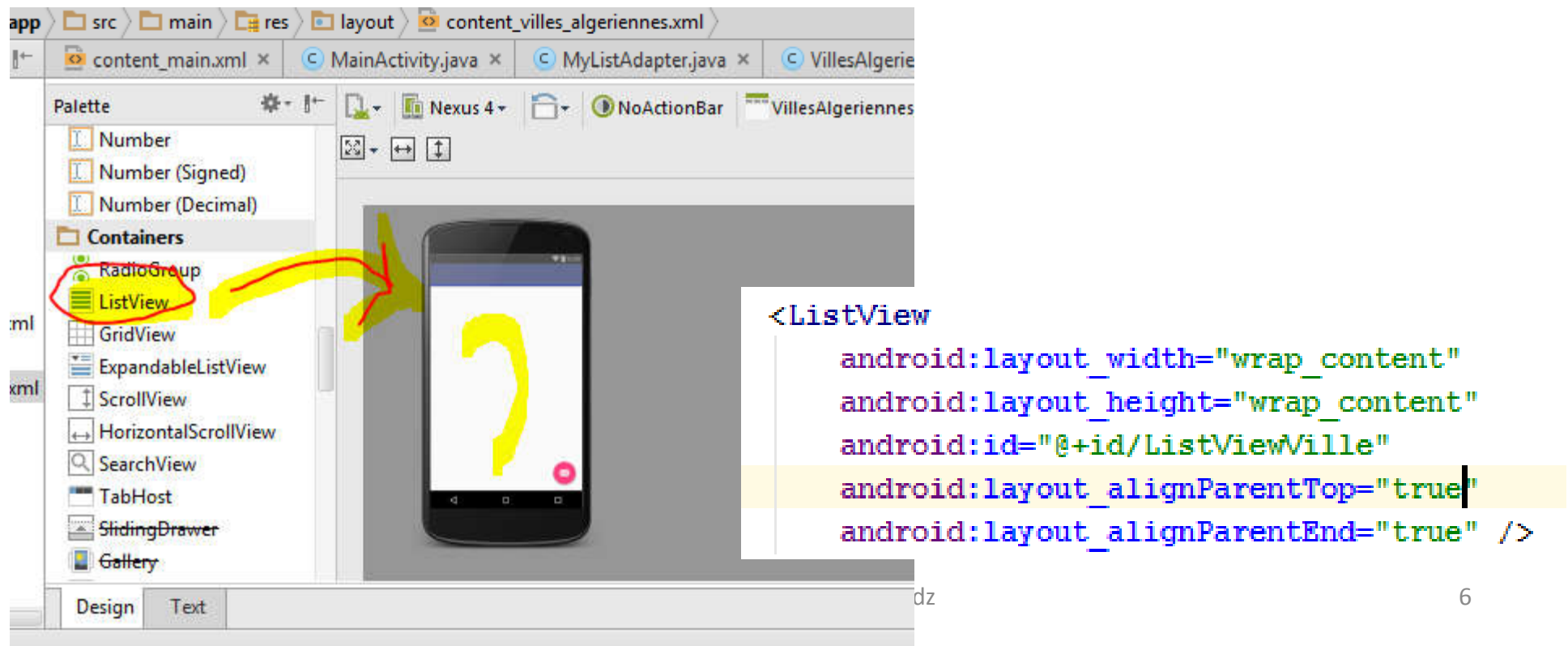


# Le composant ListView

- ListView est un composant d'interface qui permet d'afficher et manipuler une liste d'éléments.
- Les éléments peuvent être des types primitifs ou bien des Objets java.

# Exemple 1: liste des villes algériennes

- On souhaite réaliser un écran qui affiche à l'utilisateur la liste des villes algériennes dans un ListView.

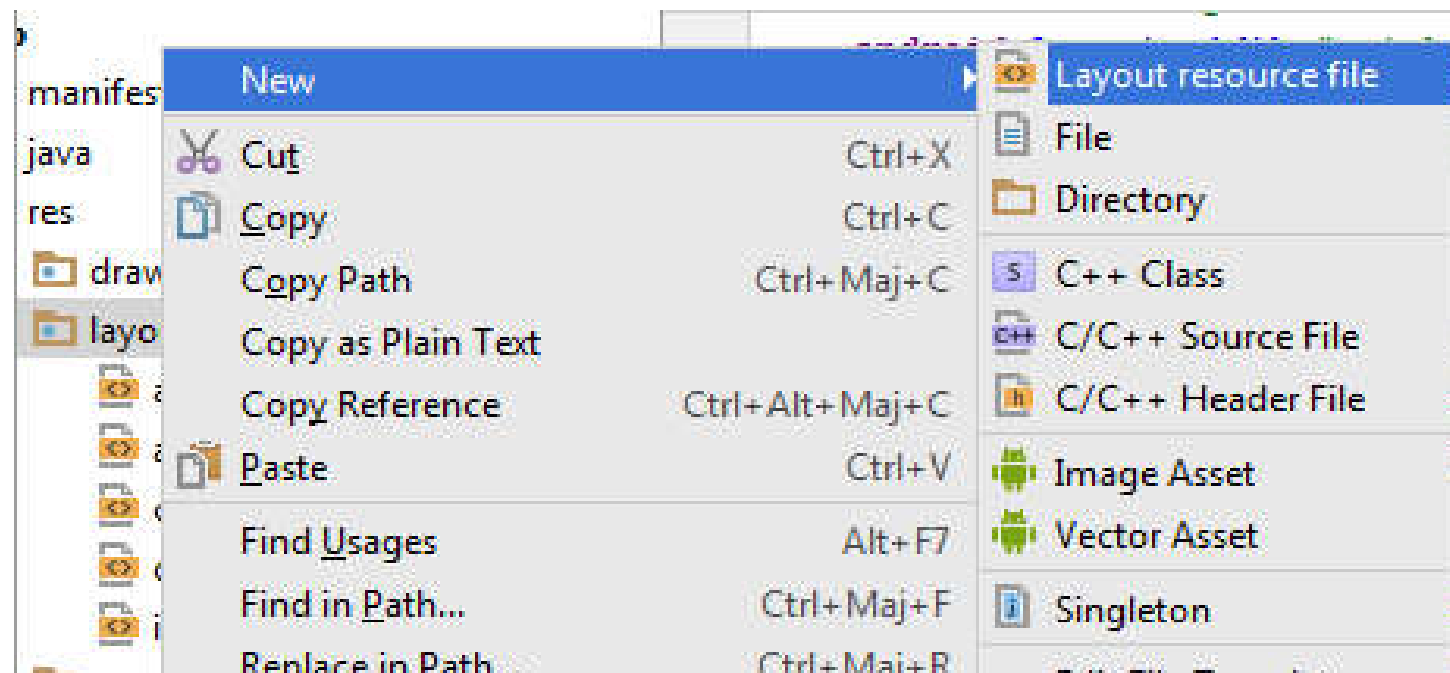


The screenshot shows the Android Studio IDE with the design view of a ListView widget on a smartphone. The Palette on the left shows the ListView widget selected. A yellow arrow points from the ListView widget in the Palette to the design view. A yellow highlight is on the XML code snippet for the ListView.

```
<ListView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:id="@+id/ListViewVille"  
    android:layout_alignParentTop="true"  
    android:layout_alignParentEnd="true" />
```

# Exemple 1: liste des villes algériennes

- Le ListView a besoin d'un modèle dans lequel un élément peut être affiché ( servira comme modèle)
  - Créer un fichier layout xml, comme suit :



# Exemple 1: liste des villes algériennes

- Dans le fichier de layout ( nommé ici **item\_ville\_layout.xml**), on définit comment un élément va être affiché (exemple : le nom des villes sera mis dans un **TextView**).

```
<?xml version="1.0" encoding="utf-8"?>
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="40dp"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="true"
    android:id="@+id/txtNomVille"/>
```



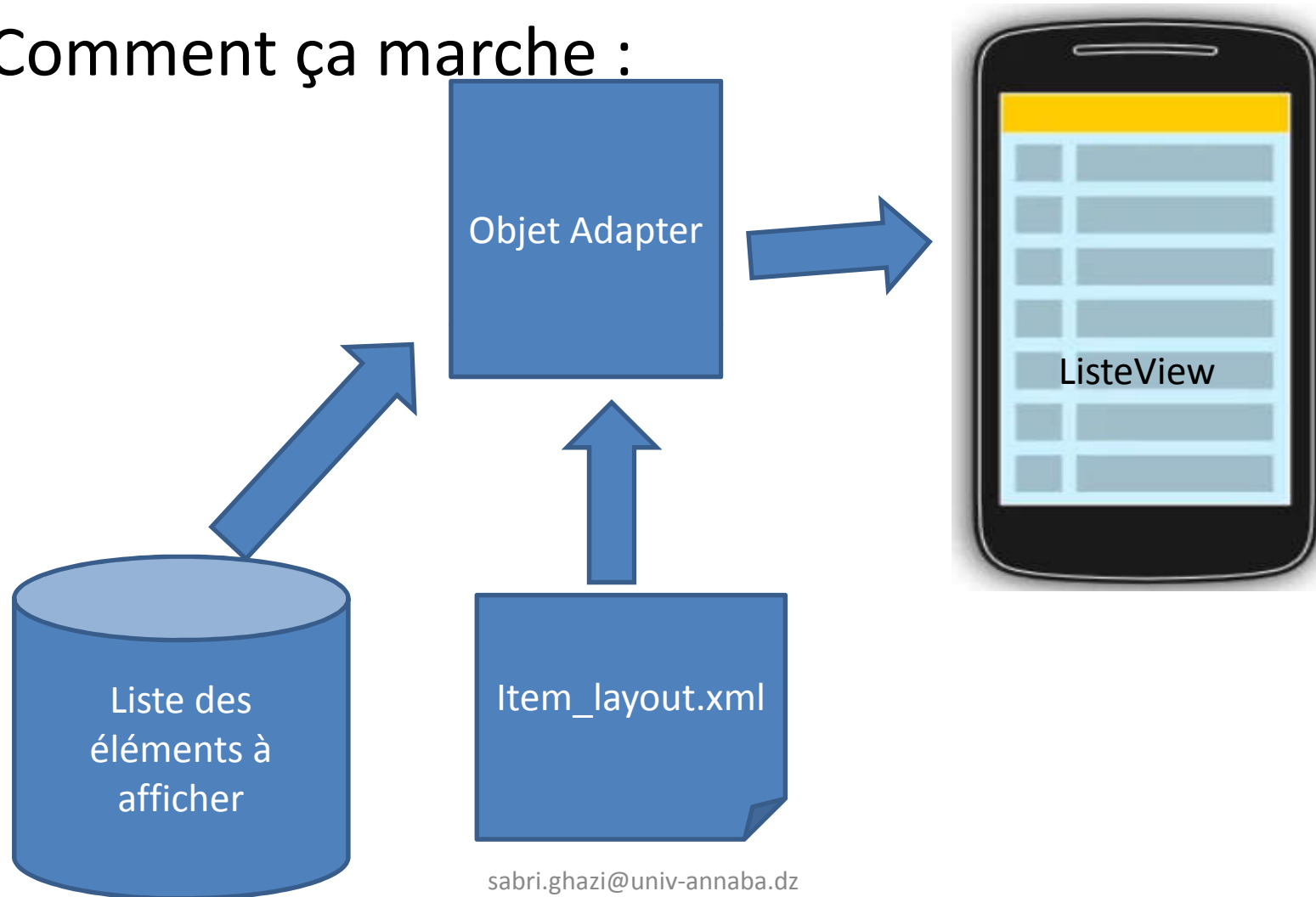
# Exemple 1: liste des villes algériennes

- Maintenant, dans le code de l'activity:
  - On définit un **Adapter**, qui permet de remplir le ListView avec une liste de valeur.
  - Cet Adapter est une classe java prédéfinie est qui hérite de la classe BaseAdapter:
    - ArrayAdapter



# Exemple 1: liste des villes algériennes

- Comment ça marche :



# L'adapter ?

## Utiliser le bon adaptateur

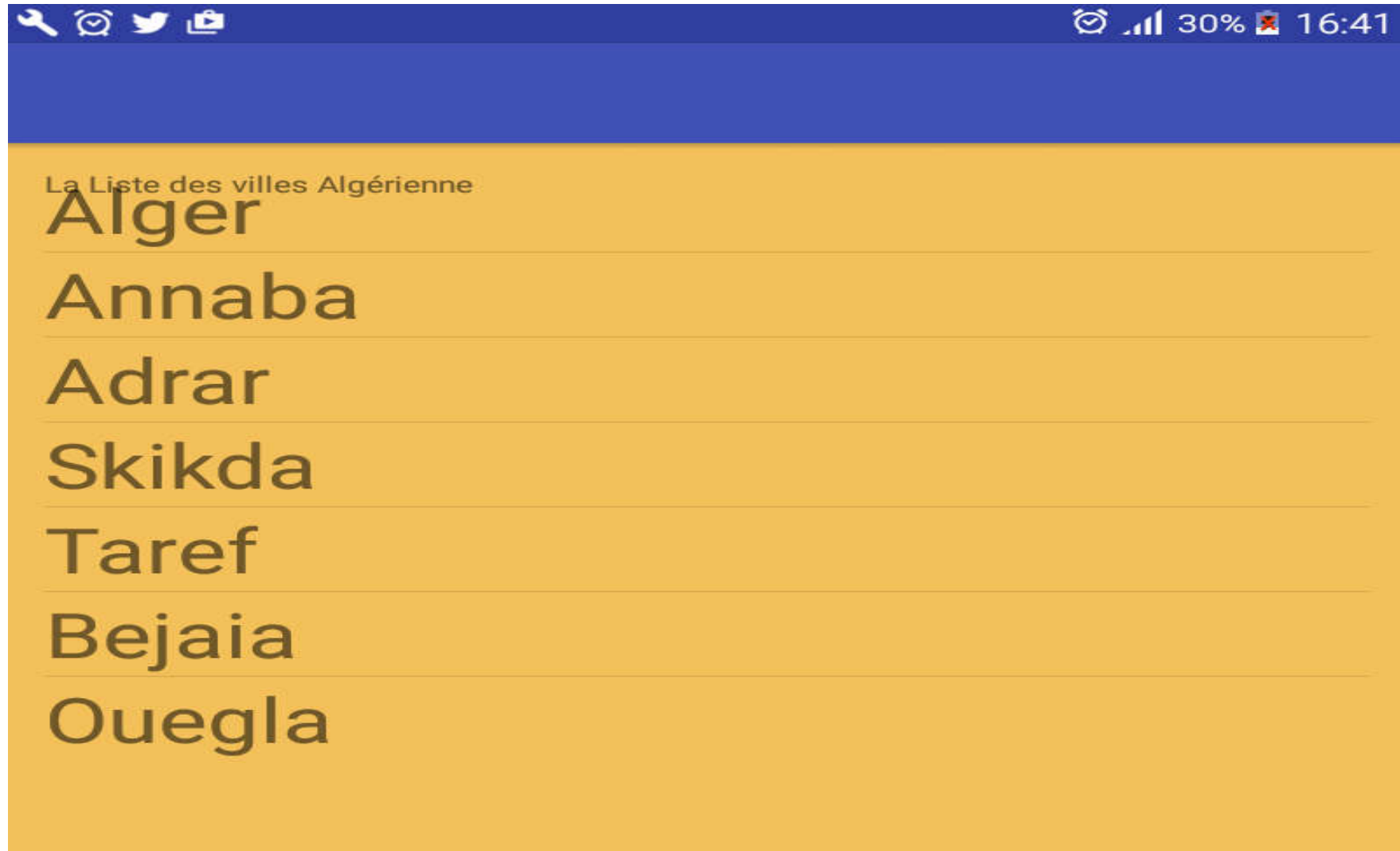
Puisque `Adapter` est une interface, il n'existe pas d'objet `Adapter` utilisable en l'état. La raison est qu'il pourrait exister autant de type d'adaptateurs qu'il existe de type de données. Bien évidemment, la plate-forme Android contient plusieurs implémentations permettant de traiter les types les plus courants :

- `ArrayAdapter<T>` : pour tous les types de tableaux ;
- `BaseAdapter` : pour créer des adaptateurs personnalisés ;
- `CursorAdapter` : pour traiter les données de type `Cursor` ;
- `HeaderViewListAdapter` : pour ajouter des entêtes et pieds de page aux `ListView` ;
- `ResourceCursorAdapter` : pour la création de vues à partir d'une disposition XML ;
- `SimpleAdapter` : malgré son nom, cet adaptateur s'emploie pour des données complexes ;
- `SimpleCursorAdapter` : sur-couche du `CursorAdapter` permettant de lier un modèle XML aux données.

# Exemple 1: liste des villes algériennes

```
//Récupérer une référence sur le composant ListView
ListView lV= (ListView)findViewById(R.id.ListViewVille);
//Le tableau qu'on cherche à afficher dans la liste
String[] Villes={"Alger",
                "Annaba", "Adrar",
                "Skikda", "Taref",
                "Bejaia", "Ouegla"};
//Création d'un Adapter,
ArrayAdapter<String> villeAdapter= new ArrayAdapter<String>(
    getBaseContext(),R.layout.item_ville_layout,Villes);
//Affecter l'adapter à la ListView.
lV.setAdapter(villeAdapter);
```

# Exemple 1 : liste des villes algériennes



## Exemple 2: afficher la liste des villes avec leur code postal

- Deux informations à afficher,
- On a une liste d'objet, chaque objet représente une ville (NomVille, CodeVille).
- Que faire dans ce cas ?
  - Écrire une classe **Ville.java**
  - Modifier le fichier **item\_ville\_layout.xml** (**on doit mettre 2 TextView**)
  - Utiliser un adapter personnalisé, c'est-à-dire écrire une classe qui hérite de la classe **ArrayAdapter**.

# Exemple 2: liste des villes et leurs code postal

- La classe **Ville.java**

```
1 public class Ville {
2
3     String nom;
4     String code;
5
6     public String getNom() {
7         return nom;
8     }
9
10    public void setNom(String nom) {
11        this.nom = nom;
12    }
13
14    public String getCode() {
15        return code;
16    }
17
18    public void setCode(String code) {
19        this.code = code;
20    }
21 }
```



# Exemple 2: liste des villes et leurs code postal

Le fichier de layout pour chaque élément

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    android:orientation="horizontal"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_height="wrap_content" android:layout_width="wrap_content">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="40dp"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:id="@+id/txtNomVille"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="40dp"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:id="@+id/txtCodeVille"/>
</LinearLayout>
```

# ListeDeVilleAdapter.java

```
public class ListeDeVilleAdapter extends ArrayAdapter<Ville>{

    public ListeDeVilleAdapter(Context context, int resource, ArrayList<Ville> objects) {
        super(context, resource, objects);
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        View v = convertView;
        if (v == null) {
            LayoutInflater vi;
            vi = LayoutInflater.from(getContext());
            v = vi.inflate(R.layout.item_ville_layout, null);
        }

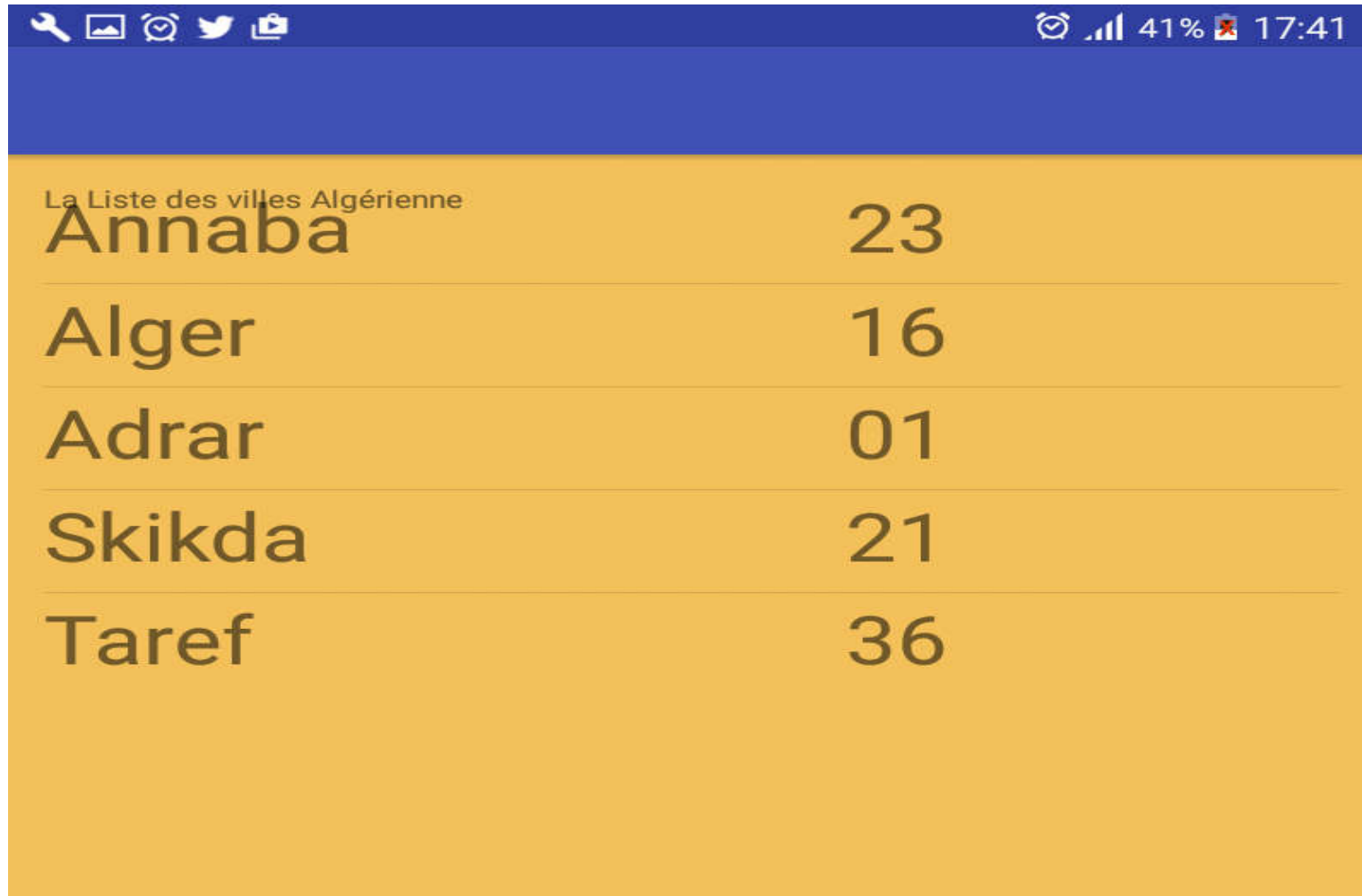
        Ville vil = getItem(position);

        if (vil != null) {
            TextView tt1 = (TextView) v.findViewById(R.id.txtNomVille);
            TextView tt2 = (TextView) v.findViewById(R.id.txtCodeVille);
            tt1.setText(vil.getNom());
            tt2.setText(vil.getCode());
        }
        return v;}
}
```

# Dans la classe de l'activité :

```
1 //Récupérer une référence sur le composant ListView
2 ListView lV= (ListView)findViewById(R.id.ListViewVille);
3 //La liste
4 ArrayList<Ville> V=new ArrayList<Ville>();
5 Ville obj= new Ville();
6 obj.setCode("23");obj.setNom("Annaba");
7 Ville obj1= new Ville();
8 obj1.setCode("16");obj1.setNom("Alger");
9 Ville obj2= new Ville();
10 obj2.setCode("01");obj2.setNom("Adrar");
11 Ville obj3= new Ville();
12 obj3.setCode("21");obj3.setNom("Skikda");
13 Ville obj4= new Ville();
14 obj4.setCode("36");obj4.setNom("Taref");
15 V.add(obj);V.add(obj1);V.add(obj2);V.add(obj3);V.add(obj4);
16 //Création d'un Adapter,
17 ListeDeVilleAdapter<Ville> villeAdapter= new ListeDeVilleAdapter<Ville>(
18     getContext(),R.layout.item_ville_layout,V);
19 //Affecter l'adapter à la ListView.
20 lV.setAdapter(villeAdapter);
21 }
```

# Exemple 2 liste des villes avec leurs code postal



The image shows a mobile application interface with a blue header bar. Below the header, the text "La Liste des villes Algérienne" is displayed. The main content area has a yellow background and contains a list of cities and their postal codes, separated by horizontal lines. The cities listed are Annaba, Alger, Adrar, Skikda, and Taref.

City	Postal Code
Annaba	23
Alger	16
Adrar	01
Skikda	21
Taref	36

# D'autres options dynamiques

- Rendre notre ListView interactif:
  - Comment trier les éléments ?
  - Comment ajouter des boutons pour faire des actions sur les éléments de la liste

# 1- Trier les éléments d'un ListView

- Modifier la classe Ville.java, implémentation de l'interface **Comparable**

```
public class Ville implements Comparable {  
  
    //....  
    //....  
    //....  
  
    @Override  
    public int compareTo(Object another) {  
        Ville o=(Ville) another;  
  
        return o.getCode().compareTo(this.getCode());  
    }  
}
```

# 1- Trier les éléments d'un ListView

- Modifier l'Adapter.

```
public class ListeDeVilleAdapter<V> extends ArrayAdapter<Ville> implements Comparator{
    public ListeDeVilleAdapter(Context context, int resource, ArrayList<Ville> objects) {
        super(context, resource, objects);
    }
    @Override
    public void sort(Comparator<? super Ville> comparator) {
        super.sort(comparator);
    }
    //.....
    //.....
}
    @Override
    public int compare(Object lhs, Object rhs) {
        Ville v1=(Ville) lhs;
        Ville v2=(Ville )rhs;
        return v1.compareTo(v2);
    }
}
```

# 1- Trier les éléments d'un ListView

- Dans le bouton qui lance le trie on fait:

```
public void Tier(View v) {  
    //..  
    ListeDeVilleAdapter adpt=(ListeDeVilleAdapter)lv.getAdapter();  
    adpt.sort(adpt);  
    //..  
}
```



PLUS GRAND

PLUS PETIT

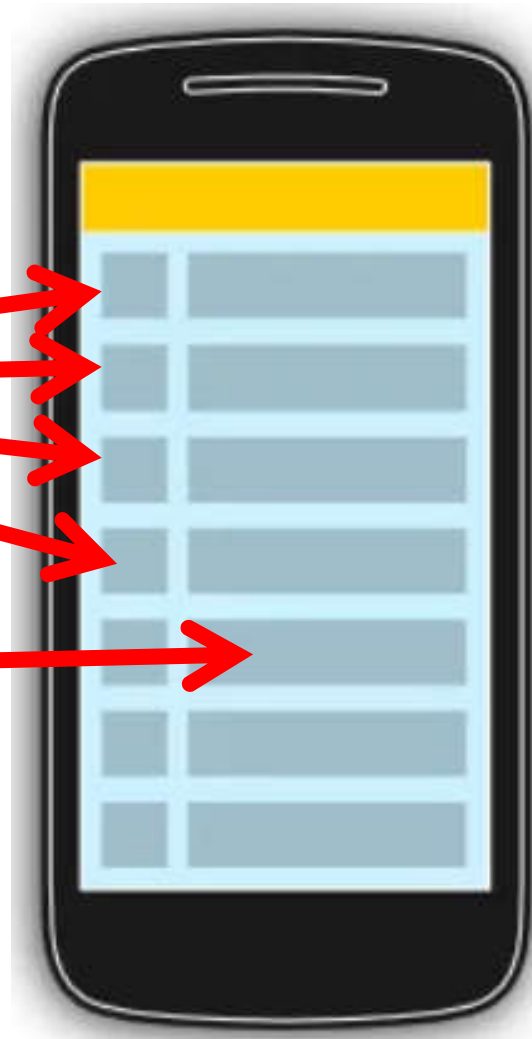
Adrar	01
Alger	16
Skikda	21
Annaba	23
Taref	36

# ListView avec des boutons

- Offrir a l'utilisateur la possibilité de modifier un élément de la liste.

Boutons

TextView



## ListViewExempleDMGHAZI

NEW BUTTON

23	Annaba	CLIQUER DOLP
16	Alger	CLIQUER DOLP
36	Taref	CLIQUER DOLP
41	Souk Ahras	CLIQUER DOLP
24	Guelma	CLIQUER DOLP
06	Bejaia	CLIQUER DOLP
18	Jijel	CLIQUER DOLP

V

**implements** View.OnClickListener

```
if(btnAction!=null) {  
    btnAction.setTag(p.getName());  
    btnAction.setOnClickListener(this);  
}
```

@Override

```
public void onClick(View v) {  
    String str=v.getTag().toString();  
    Toast.makeText(getApplicationContext(), str, Toast.LENGTH_SHORT).show();  
}
```