android

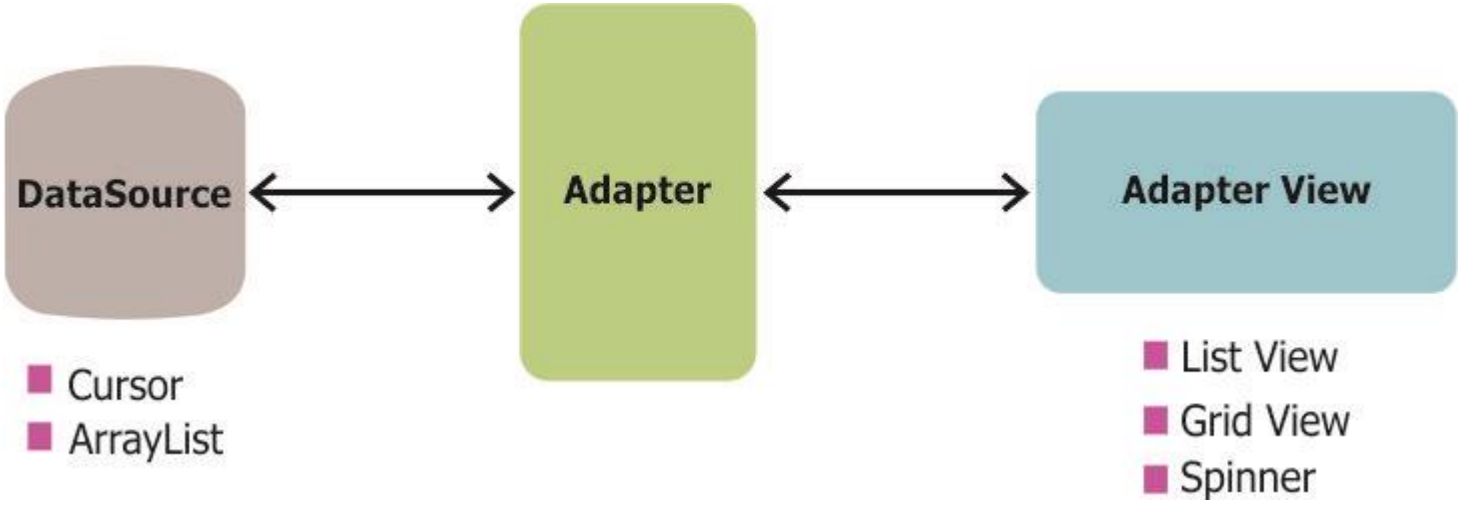
An Open Handset Alliance Project

ListView et Adapter

O.Legrand
G. Seront

ListView & GridView

- Il existe des Widgets dont le contenu est dynamique (ex: ListView)
- L'Adapter fait le pont entre la source de donnée (DB, Collection, …) et la View:



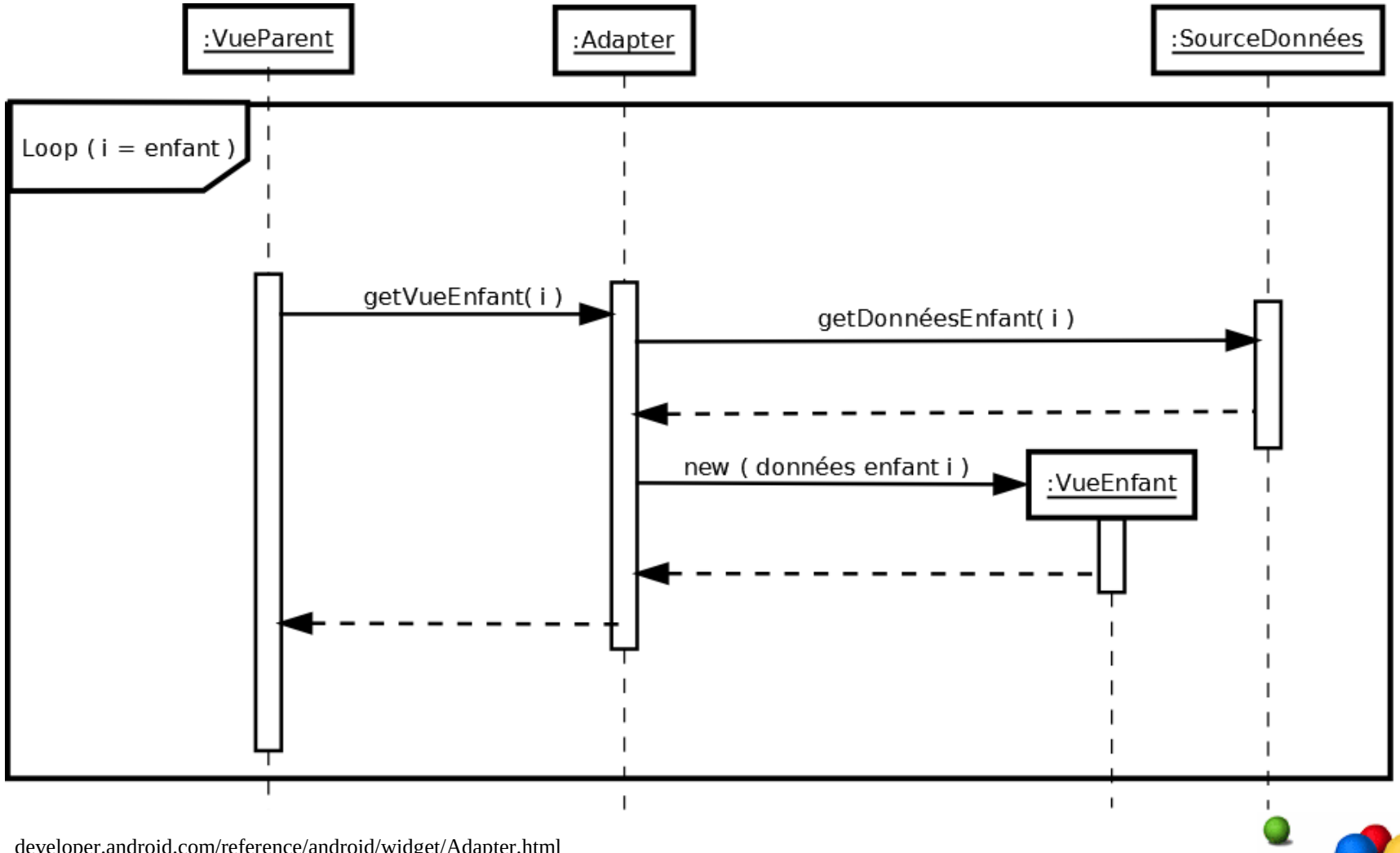
http://developer.android.com/guide/topics/ui/declaring-layout.html#AdapterViews
https://github.com/codepath/android_guides/wiki/Using-an-ArrayAdapter-with-ListView

Adapter

- Objet qui relie une vue à une source de données ;
- La vue (vue parent) affiche des vues enfant ;
- Chaque vue enfant affiche les données d'un élément de la source de données ;
- L'adapter doit, pour chaque demande de la vue parent, lui fournir une vue enfant contenant les données d'un élément de la source de données ;

developer.android.com/reference/android/widget/Adapter.html

Adapter : diagramme de séquence



developer.android.com/reference/android/widget/Adapter.html

Adapters souvent utilisés

- Les adapters suivant sont souvent utilisés :
  - *ArrayAdapter :*
    - lie un *AdapterView* affichant une liste de données
    - à un tableau contenant ces données
  - *SimpleCursorAdapter :*
    - lie un *AdapterView* affichant des colonnes de données
    - à un *ContentProvider* (ex: base de données)

developer.android.com/reference/android/widget/Adapter.html

ArrayAdapter

- Un *ArrayAdapter*
  - lie un *AdapterView* à un tableau de données ;
- L' *AdapterView*, la vue parent
  - affiche une liste de vues enfant (ex: *ListView*)
- Chaque vue enfant
  - affiche le *toString()* d'un objet du tableau ;

developer.android.com/reference/android/widget/Adapter.html

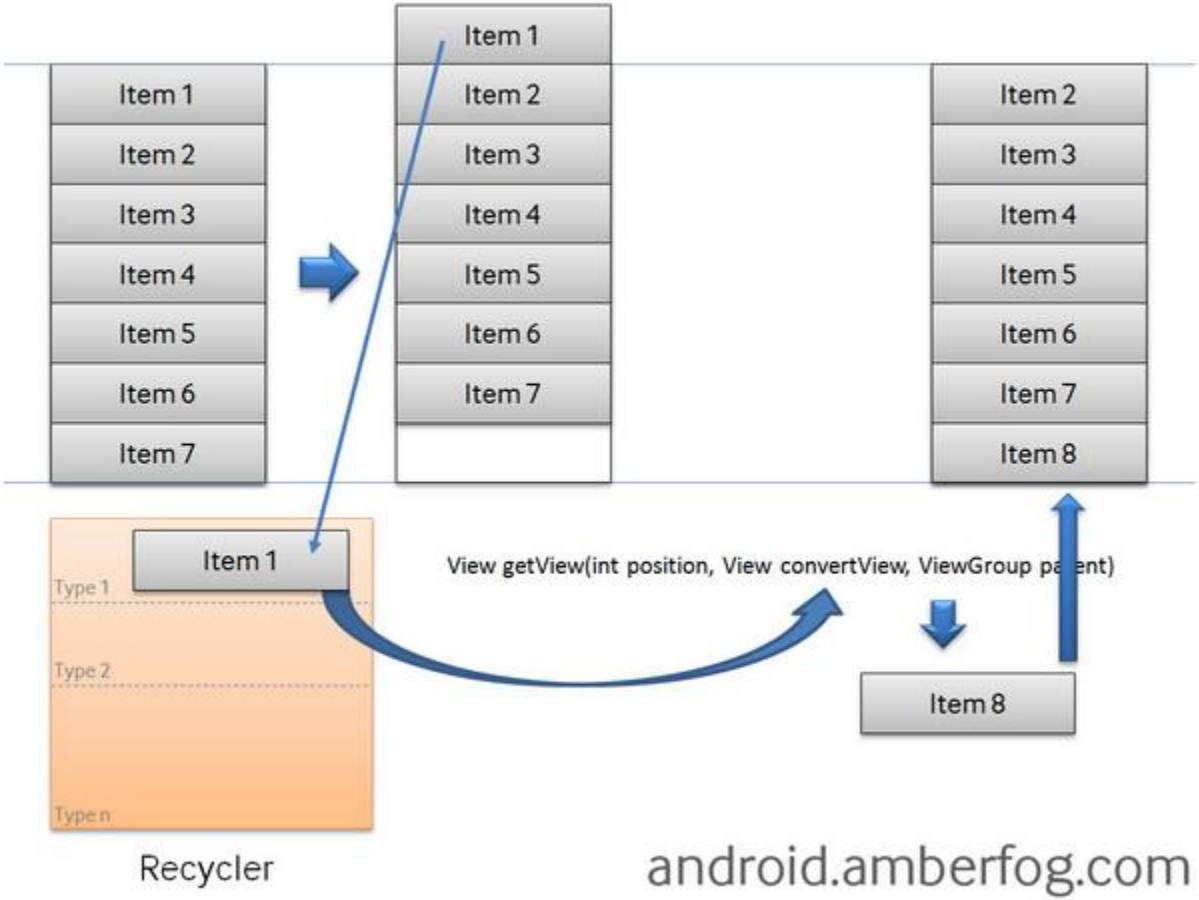
ArrayAdapter

- La vue parent, avant de s'afficher :
  - demande à l'adapter de lui construire, une par une, les vues enfant ;
  - en appelant sur l'adapter la méthode *getView(...)*
- L'adapter
  - est responsable de la création des vues enfant qu'il doit fournir à la vue parent ;
  - affecte à chaque vue enfant qu'il crée, le *toString()* de l'objet correspondant dans le tableau de données;

=>Le résultat à l'affichage est limité à ce *toString()*

developer.android.com/reference/android/widget/Adapter.html

Recyclage des vues



developer.android.com/reference/android/widget/Adapter.html

ArrayAdapter

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

Layout des éléments de la liste

```
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@android:id/text1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceListItemSmall"
    android:gravity="center_vertical"
    android:paddingStart="?android:attr/listPreferredItemPaddingStart"
    android:paddingEnd="?android:attr/listPreferredItemPaddingEnd"
    android:minHeight="?android:attr/listPreferredItemHeightSmall" />
```

https://github.com/codepath/android_guides/wiki/Using-an-ArrayAdapter-with-ListView

ArrayAdapter

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

Layout des éléments de la liste

```
ListView listView = (ListView) findViewById(R.id.listview);
listView.setAdapter(adapter);
```

https://github.com/codepath/android_guides/wiki/Using-an-ArrayAdapter-with-ListView

CursorAdapter

Colonnes à extraires

```
String[] fromColumns = {ContactsContract.Data.DISPLAY_NAME,
                        ContactsContract.CommonDataKinds.Phone.NUMBER};
int[] toViews = {R.id.display_name, R.id.phone_number};
```

Id des TextView dans le layout person_name_and_number

```
SimpleCursorAdapter adapter = new SimpleCursorAdapter(this,
        R.layout.person_name_and_number, cursor, fromColumns, toViews, 0);
ListView listView = getListView();
listView.setAdapter(adapter);
```

Layout des éléments de la liste

https://github.com/codepath/android_guides/wiki/Using-an-ArrayAdapter-with-ListView

ArrayAdapter

- Si la source de données est modifiée :
  - Il faut prévenir l'adapter ;
  - car la vue parent doit refléter ces changements ;
- Appel de la méthode :
  - *adapter.notifyDataSetChanged()*

developer.android.com/reference/android/widget/Adapter.html

Personnaliser l'ArrayAdapter

- Pour améliorer l'affichage :
  - Définir un layout qui représente un élément de la liste
  - Écrire une classe qui étend *ArrayAdapter*
  - Redéfinir sa méthode *getView(...)* afin d'assigner les propriétés de l'objet aux vues qui composent la vue enfant.

developer.android.com/reference/android/widget/Adapter.html

Personnaliser l'ArrayAdapter

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
 android:layout_width="match_parent"
 android:layout_height="match_parent" >
    <TextView
      android:id="@+id/tvName"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Name" />
   <TextView
      android:id="@+id/tvHome"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="HomeTown" />
</LinearLayout>
```

https://github.com/codepath/android_guides/wiki/Using-an-ArrayAdapter-with-ListView

Personnaliser l'ArrayAdapter

```
public class UsersAdapter extends ArrayAdapter<User> {
    public UsersAdapter(Context context, ArrayList<User> users) {
       super(context, 0, users);
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
       // Get the data item for this position
       User user = getItem(position);
       // Check if an existing view is being reused, otherwise inflate the view
       if (convertView == null) {
          convertView = LayoutInflater.from(getContext()).inflate(R.layout.item_user,
       }
       // Lookup view for data population
       TextView tvName = (TextView) convertView.findViewById(R.id.tvName);
       TextView tvHome = (TextView) convertView.findViewById(R.id.tvHome);
       // Populate the data into the template view using the data object
       tvName.setText(user.name);
       tvHome.setText(user.hometown);
       // Return the completed view to render on screen
       return convertView;
   }
}
```

https://github.com/codepath/android_guides/wiki/Using-an-ArrayAdapter-with-ListView

Méthode *getView(...)*

- La méthode *getView(...)* :
  - est responsable de :
    - construire chaque vue enfant ;
    - lui assigner les propriétés de l'objet qu'il représente ;
    - mémoriser la position de l'objet courant.
  - appelée par la vue parent pour chaque enfant ;
  - paramètres reçus :
    - position dans la liste de la vue parent ;
    - référence d'une vue enfant. Si pas null, ne pas créer la vue enfant. Modifier ses attributs (pour améliorer les performances);
    - référence vue parent (peu utilisée).

developer.android.com/reference/android/widget/Adapter.html