Programmation d'application mobiles *Android*

Vues Dynamiques - Adapter

C. Raïevsky

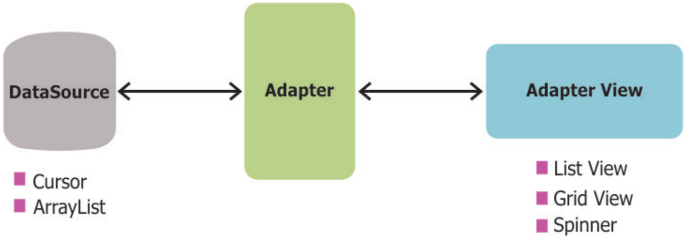
Avec la courtoisie de S. Jean

UNIVERSITÉ Grenoble Alpes IUT Valence

Département Informatique

ViewGroup dynamiques et Adapter

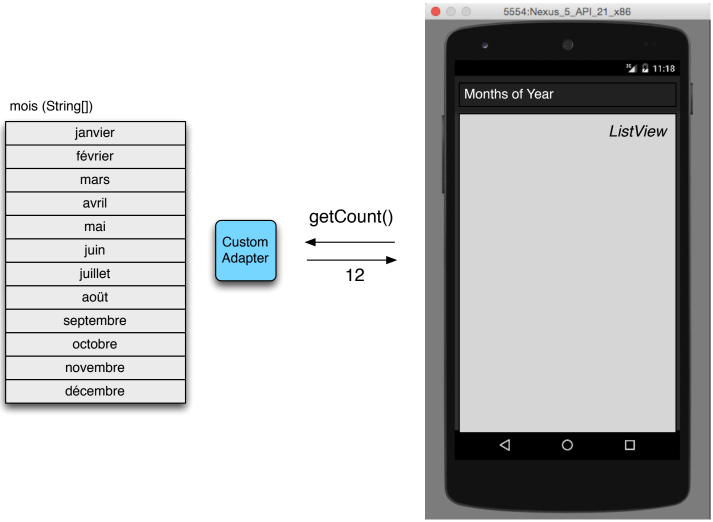
- **Groupe de vues dynamique**
  - **Nombre variable de vues**, non connu à priori
  - **`Spinner`**, **`ListView`**, **`GridView`**, **`RecyclerView`**
- **Adapter** (interface `android.widget.Adapter`)
  - **Délivre les vues sur demande** à partir d'une **source de données**
    - **source considérée comme une liste** (position ≥ 0)



source

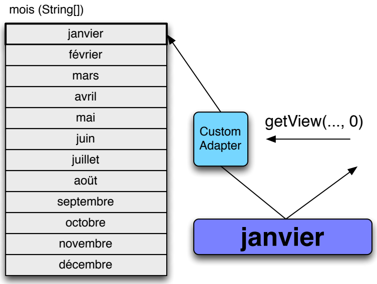
Adapter ↔ AdapterView : exemple de `ListView`

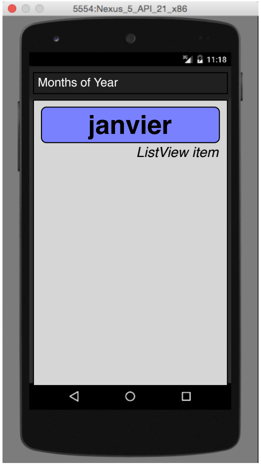
- `getCount()` : connaitre la taille de la source de données



Adapter ↔ AdapterView : exemple de `ListView`

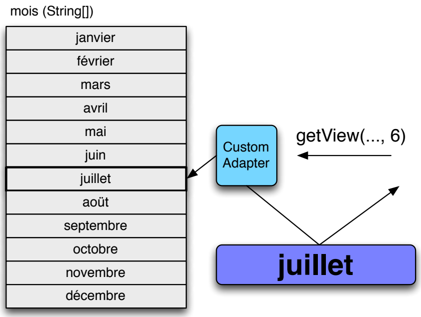
- **`getView(ViewGroup parent, View cv, int pos)`**
  - **Obtention d'une vue à partir de sa position**
  - Charge à l'*adapter* de la construire

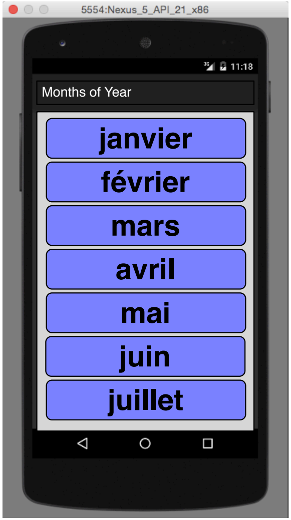




Adapter ↔ AdapterView : exemple de `ListView`

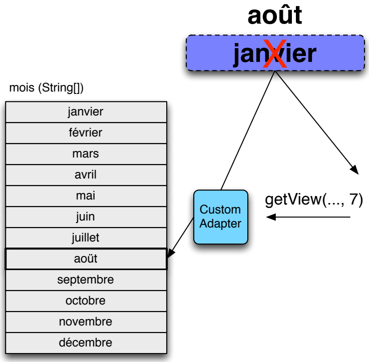
- Appel à `getView` tant qu'il est possible d'afficher les vues suivantes

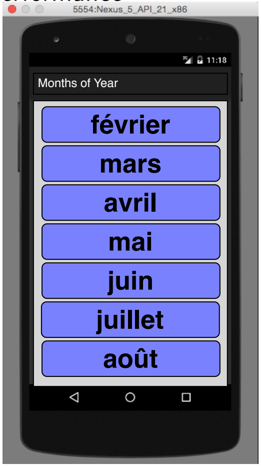




Adapter ↔ AdapterView : exemple de `ListView`

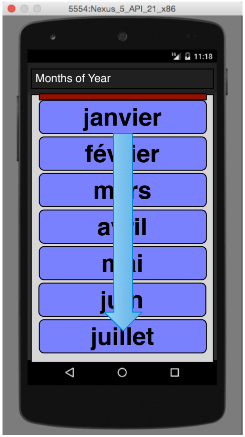
- Appel de la vue suivante lors de la disparition d'une vue par défilement
- L'ancien objet `View` est passé en paramètre pour être modifié
  - *Recyclage* pour des raisons de performance

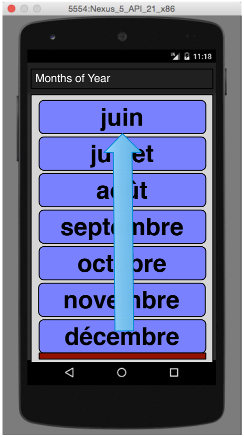




Adapter $\leftrightarrow$ AdapterView

- Gestion automatique du non défilement après la fin de liste
  - Aucun appel à l'adapter
  - Remarque : idem pour le non défilement avant le début de liste

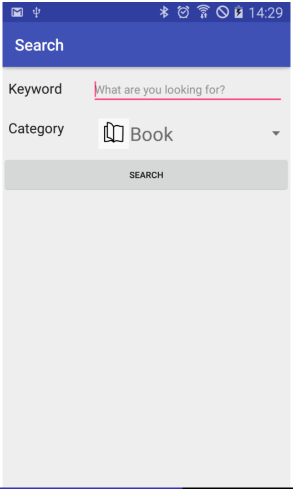




Application exemple

Recherche dans une liste d'items, par catégorie

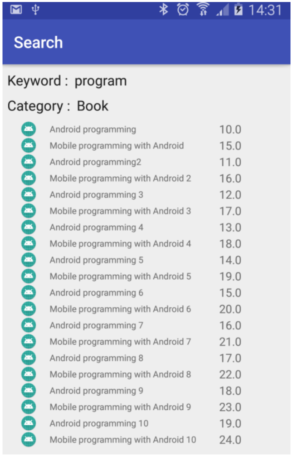
Home Activity



Keyword What are you looking for?

Category Book

Result Activity

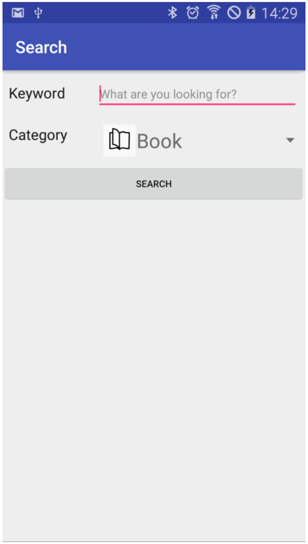


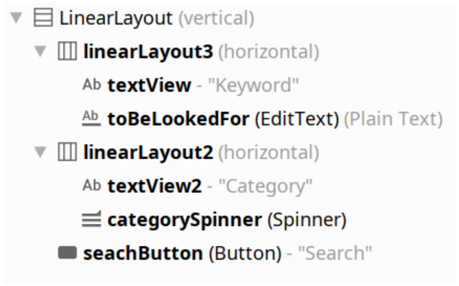
Keyword : program

Category : Book

Android programming	10.0
Mobile programming with Android	15.0
Android programming2	11.0
Mobile programming with Android 2	16.0
Android programming 3	12.0
Mobile programming with Android 3	17.0
Android programming 4	13.0
Mobile programming with Android 4	18.0
Android programming 5	14.0
Mobile programming with Android 5	19.0
Android programming 6	15.0
Mobile programming with Android 6	20.0
Android programming 7	16.0
Mobile programming with Android 7	21.0
Android programming 8	17.0
Mobile programming with Android 8	22.0
Android programming 9	18.0
Mobile programming with Android 9	23.0
Android programming 10	19.0
Mobile programming with Android 10	24.0

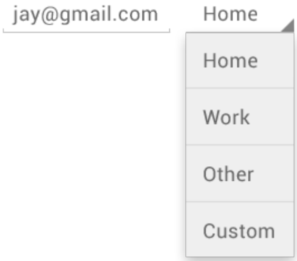
Application exemple : layout de l'activité `HomeActivity`




LinearLayout (vertical)
- **linearLayout3** (horizontal)
  - **textView** - "Keyword"
  - **toBeLookedFor** (EditText) (Plain Text)
- **linearLayout2** (horizontal)
  - **textView2** - "Category"
  - **categorySpinner** (Spinner)
- **seachButton** (Button) - "Search"

Spinner

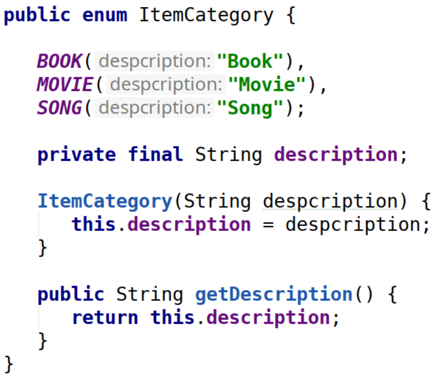
- **Liste déroulante de choix**
  - **Sélection unique**, par défaut
  - Source : `developer.android.com`



- Eléments (choix) peuplés via un **Adapter**
- **Layout personnalisable** pour les éléments

Application exemple : ItemCategory

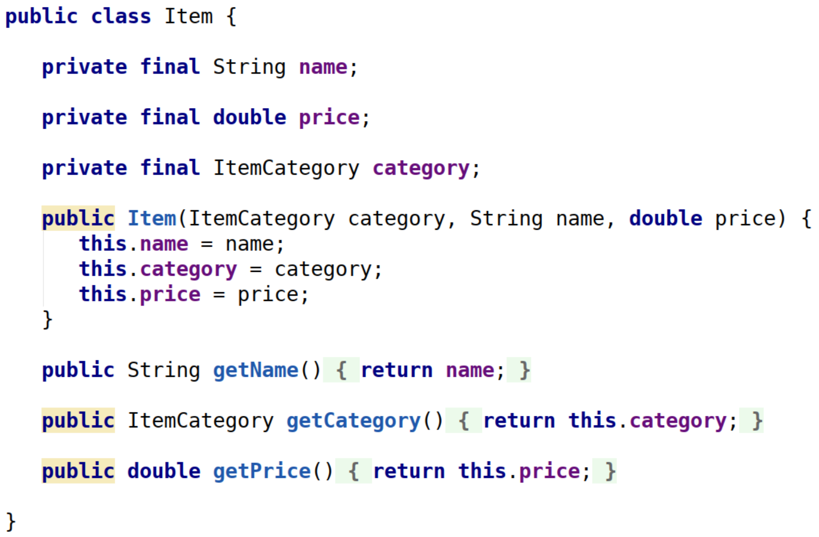
- Enumération des catégories d'objets
  - Chaque catégorie possède une description textuelle



```
public enum ItemCategory {

    BOOK( despcription: "Book"),
    MOVIE( despcription: "Movie"),
    SONG( despcription: "Song");

    private final String description;

    ItemCategory(String despcription) {
        this.description = despcription;
    }

    public String getDescription() {
        return this.description;
    }
}
```

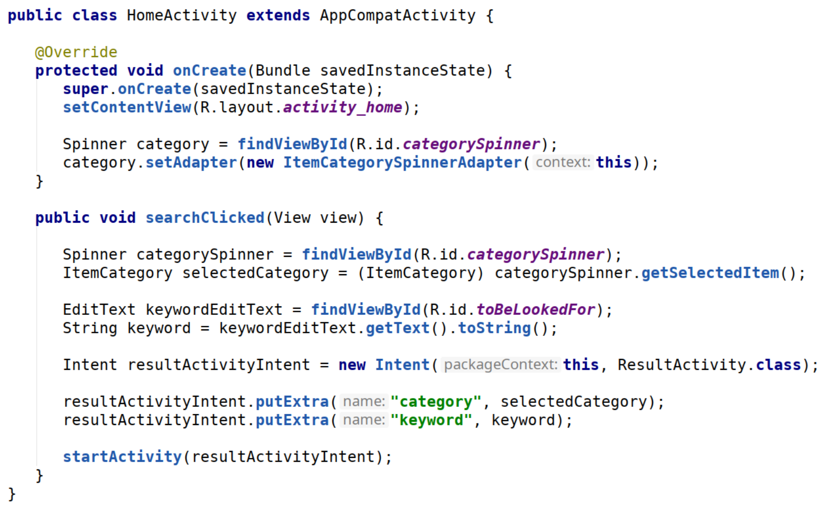
Application exemple : Item

- Modèle métier d'un objet



```java
public class Item {

    private final String name;

    private final double price;

    private final ItemCategory category;

    public Item(ItemCategory category, String name, double price) {
        this.name = name;
        this.category = category;
        this.price = price;
    }

    public String getName() { return name; }

    public ItemCategory getCategory() { return this.category; }

    public double getPrice() { return this.price; }

}
```

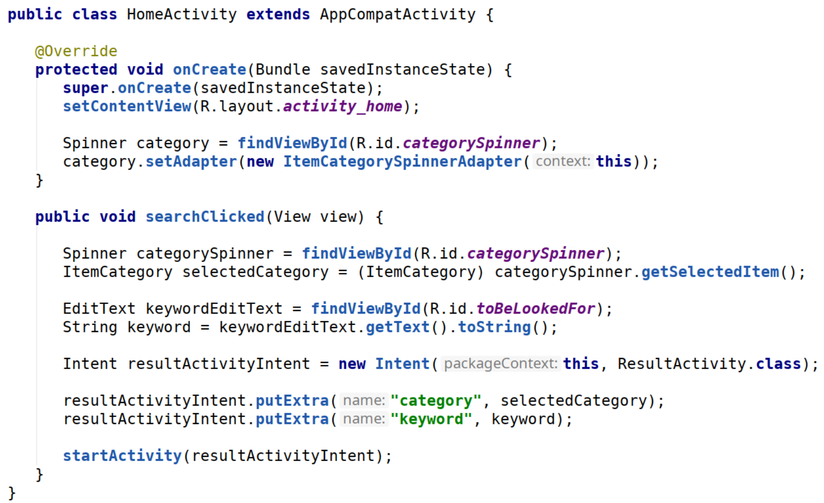
Application exemple : code de `HomeActivity`



```java
public class HomeActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);

        Spinner category = findViewById(R.id.categorySpinner);
        category.setAdapter(new ItemCategorySpinnerAdapter( context: this));
    }

    public void searchClicked(View view) {

        Spinner categorySpinner = findViewById(R.id.categorySpinner);
        ItemCategory selectedCategory = (ItemCategory) categorySpinner.getSelectedItem();

        EditText keywordEditText = findViewById(R.id.toBeLookedFor);
        String keyword = keywordEditText.getText().toString();

        Intent resultActivityIntent = new Intent( packageContext: this, ResultActivity.class);

        resultActivityIntent.putExtra( name: "category", selectedCategory);
        resultActivityIntent.putExtra( name: "keyword", keyword);

        startActivity(resultActivityIntent);
    }
}
```

- *Adapter* spécifique associé explicitement au Spinner dans `onCreate`
    - Une activité est un Context

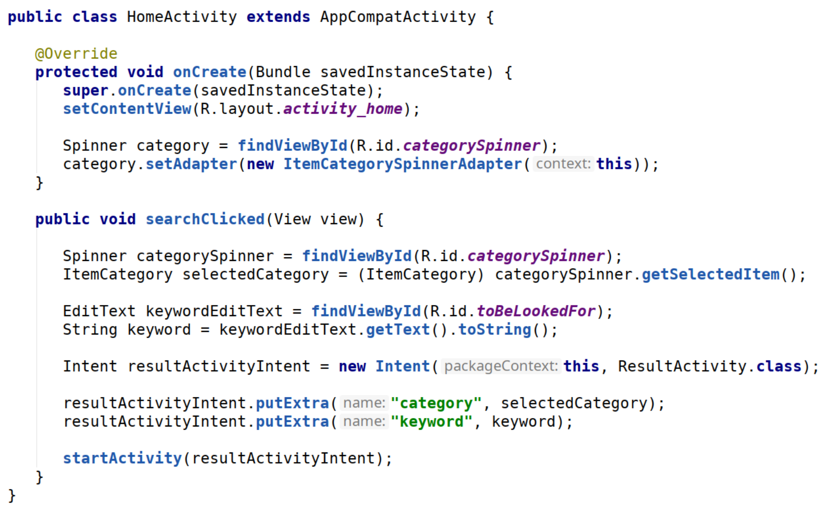
Application exemple : code de l'activité `HomeActivity`



```
public class HomeActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);

        Spinner category = findViewById(R.id.categorySpinner);
        category.setAdapter(new ItemCategorySpinnerAdapter( context: this));
    }

    public void searchClicked(View view) {

        Spinner categorySpinner = findViewById(R.id.categorySpinner);
        ItemCategory selectedCategory = (ItemCategory) categorySpinner.getSelectedItem();

        EditText keywordEditText = findViewById(R.id.toBeLookedFor);
        String keyword = keywordEditText.getText().toString();

        Intent resultActivityIntent = new Intent( packageContext: this, ResultActivity.class);

        resultActivityIntent.putExtra( name: "category", selectedCategory);
        resultActivityIntent.putExtra( name: "keyword", keyword);

        startActivity(resultActivityIntent);
    }
}
```

- *"searchClicked"* associée au bouton dans le layout :
  `android:onClick="searchClicked"`
- Lance l'activité ***ResultActivity***

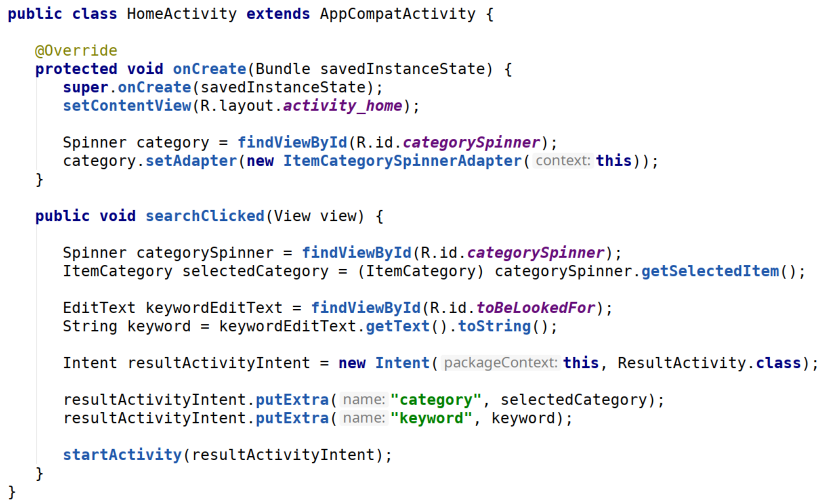
Application exemple : code de l'activité `HomeActivity`



```java
public class HomeActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);

        Spinner category = findViewById(R.id.categorySpinner);
        category.setAdapter(new ItemCategorySpinnerAdapter( context: this));
    }

    public void searchClicked(View view) {

        Spinner categorySpinner = findViewById(R.id.categorySpinner);
        ItemCategory selectedCategory = (ItemCategory) categorySpinner.getSelectedItem();

        EditText keywordEditText = findViewById(R.id.toBeLookedFor);
        String keyword = keywordEditText.getText().toString();

        Intent resultActivityIntent = new Intent( packageContext: this, ResultActivity.class);

        resultActivityIntent.putExtra( name: "category", selectedCategory);
        resultActivityIntent.putExtra( name: "keyword", keyword);

        startActivity(resultActivityIntent);
    }
}
```

- Récupération des critères de recherche
- `getText()` ne renvoie pas directement une chaîne dans le cas d'une vue `EditText` : **`toString`**

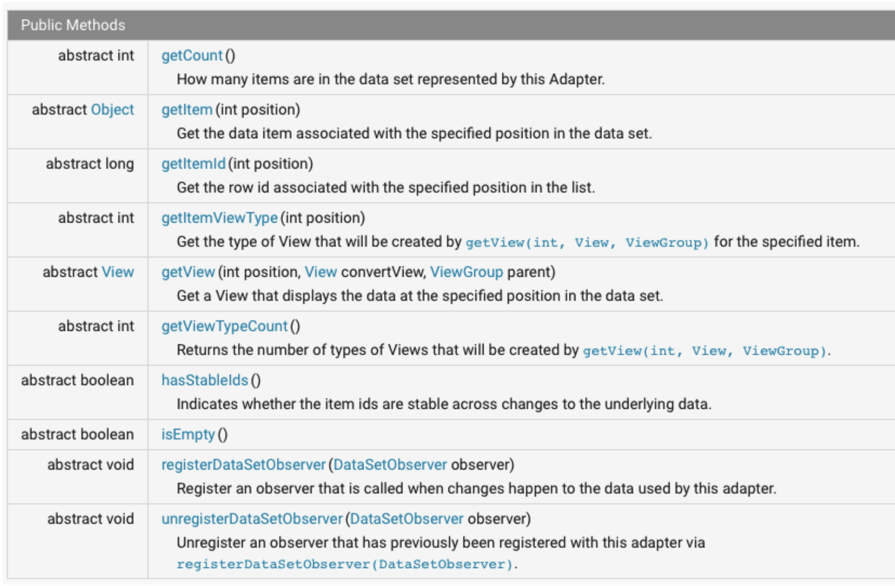
Application exemple : code de l’activité `HomeActivity`



```java
public class HomeActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);

        Spinner category = findViewById(R.id.categorySpinner);
        category.setAdapter(new ItemCategorySpinnerAdapter( context: this));
    }

    public void searchClicked(View view) {

        Spinner categorySpinner = findViewById(R.id.categorySpinner);
        ItemCategory selectedCategory = (ItemCategory) categorySpinner.getSelectedItem();

        EditText keywordEditText = findViewById(R.id.toBeLookedFor);
        String keyword = keywordEditText.getText().toString();

        Intent resultActivityIntent = new Intent( packageContext: this, ResultActivity.class);

        resultActivityIntent.putExtra( name: "category", selectedCategory);
        resultActivityIntent.putExtra( name: "keyword", keyword);

        startActivity(resultActivityIntent);
    }
}
```

- `getSelectedItem()` renvoie l’objet (*ItemCategory*) associé à l’élément du Spinner sélectionné
    - Transtypage explicite

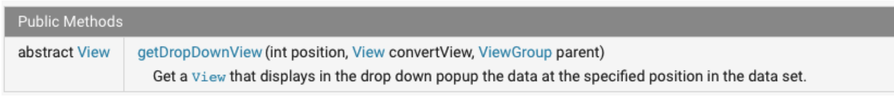
Interface `android.widget.Adapter`



Public Methods	
abstract int	getCount () How many items are in the data set represented by this Adapter.
abstract Object	getItem (int position) Get the data item associated with the specified position in the data set.
abstract long	getItemId (int position) Get the row id associated with the specified position in the list.
abstract int	getItemViewType (int position) Get the type of View that will be created by `getView(int, View, ViewGroup)` for the specified item.
abstract View	getView (int position, View convertView, ViewGroup parent) Get a View that displays the data at the specified position in the data set.
abstract int	getViewTypeCount () Returns the number of types of Views that will be created by `getView(int, View, ViewGroup)`.
abstract boolean	hasStableIds () Indicates whether the item ids are stable across changes to the underlying data.
abstract boolean	isEmpty ()
abstract void	registerDataSetObserver (DataSetObserver observer) Register an observer that is called when changes happen to the data used by this adapter.
abstract void	unregisterDataSetObserver (DataSetObserver observer) Unregister an observer that has previously been registered with this adapter via `registerDataSetObserver(DataSetObserver)`.

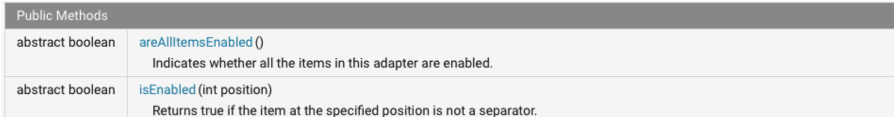
Interfaces SpinnerAdapter et ListAdapter

- Interface **SpinnerAdapter** (extension de Adapter)



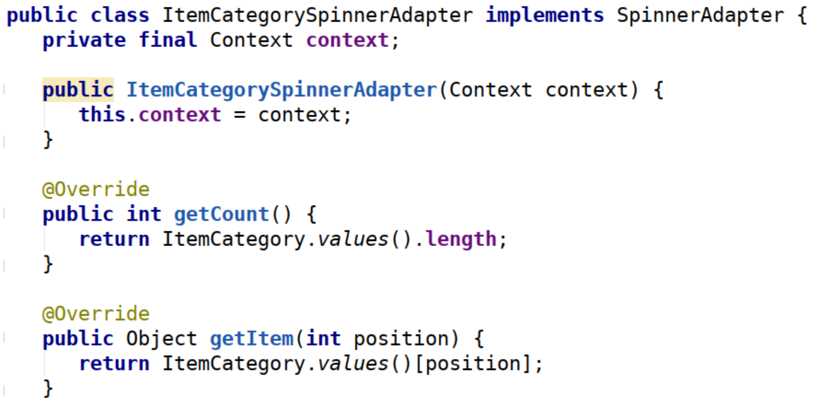
Public Methods	
abstract View	getDropDownView (int position, View convertView, ViewGroup parent) Get a View that displays in the drop down popup the data at the specified position in the data set.

- Interface **ListAdapter** (extension de Adapter)



Public Methods	
abstract boolean	areAllItemsEnabled () Indicates whether all the items in this adapter are enabled.
abstract boolean	isEnabled (int position) Returns true if the item at the specified position is not a separator.

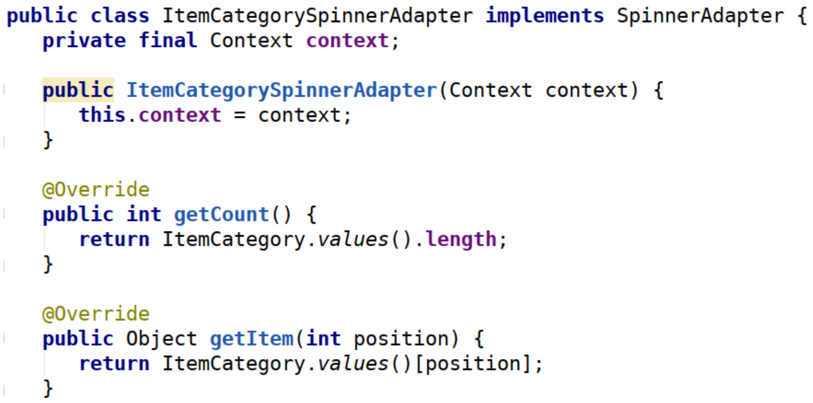
Application exemple : ItemCategorySpinnerAdapter



```java
public class ItemCategorySpinnerAdapter implements SpinnerAdapter {
   private final Context context;

   public ItemCategorySpinnerAdapter(Context context) {
      this.context = context;
   }

   @Override
   public int getCount() {
      return ItemCategory.values().length;
   }

   @Override
   public Object getItem(int position) {
      return ItemCategory.values()[position];
   }
```

- Implementation de l'interface `SpinnerAdapter`
- Constructeur prenant en paramètre un **contexte**
    - Type `android.content.Context`

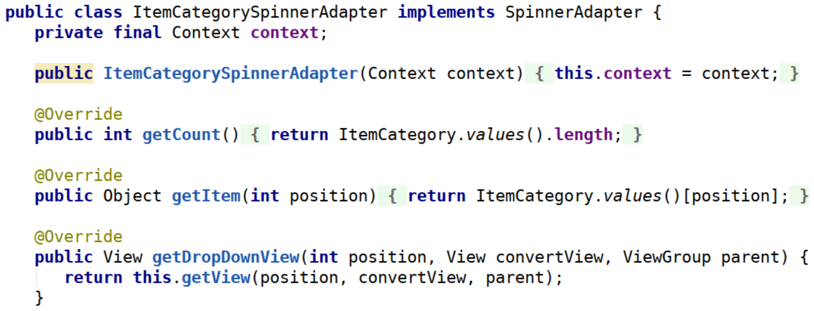
Application exemple : `ItemCategorySpinnerAdapter`



```java
public class ItemCategorySpinnerAdapter implements SpinnerAdapter {
   private final Context context;

   public ItemCategorySpinnerAdapter(Context context) {
      this.context = context;
   }

   @Override
   public int getCount() {
      return ItemCategory.values().length;
   }

   @Override
   public Object getItem(int position) {
      return ItemCategory.values()[position];
   }
```

- Implémentation de **getCount** → taille de l'énumération
- Implémentation de **getItem** → déréférencement du tableau
  - Remarque : appelé par `getSelectedItem()`

Application exemple : ItemCategorySpinnerAdapter



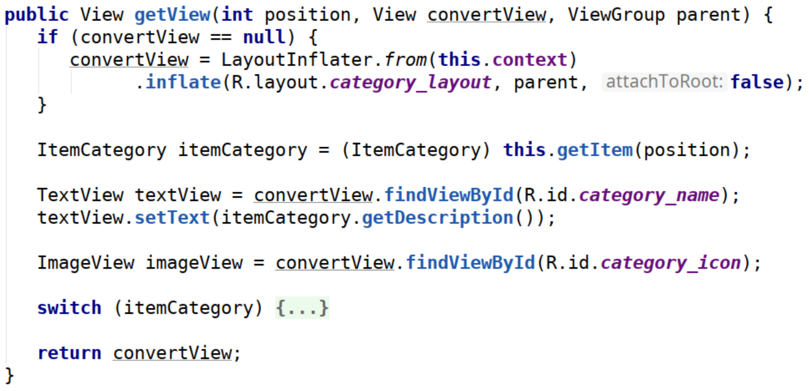
```
public class ItemCategorySpinnerAdapter implements SpinnerAdapter {
   private final Context context;

   public ItemCategorySpinnerAdapter(Context context) { this.context = context; }

   @Override
   public int getCount() { return ItemCategory.values().length; }

   @Override
   public Object getItem(int position) { return ItemCategory.values()[position]; }

   @Override
   public View getDropDownView(int position, View convertView, ViewGroup parent) {
      return this.getView(position, convertView, parent);
   }
```

- Implémentation de **getDropDownView** → indirection vers getView

Application exemple : `ItemCategorySpinnerAdapter`

- **Implémentation de `getView`**
  - Création (si besoin) d'une **nouvelle vue**
    - Par **installation du layout correspondant**
  - **Obtention de l'objet** (issu de la source de données)
    - En **fonction de la position**
  - **Remplissage de la vue**
    - Avec les **informations associées à l'objet**
  - **Retour de la référence de la vue**

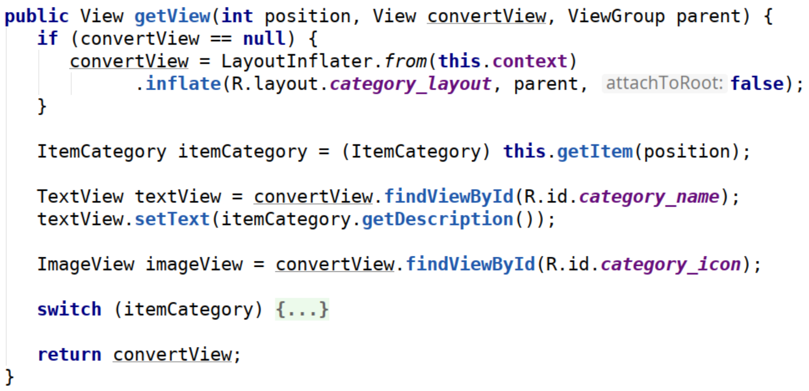
Application exemple : ItemCategorySpinnerAdapter



```
public View getView(int position, View convertView, ViewGroup parent) {
    if (convertView == null) {
        convertView = LayoutInflater.from(this.context)
                .inflate(R.layout.category_layout, parent,  attachToRoot: false);
    }

    ItemCategory itemCategory = (ItemCategory) this.getItem(position);

    TextView textView = convertView.findViewById(R.id.category_name);
    textView.setText(itemCategory.getDescription());

    ImageView imageView = convertView.findViewById(R.id.category_icon);

    switch (itemCategory) {...}

    return convertView;
}
```

- Obtention d'un **installateur de layout** (`LayoutInflater`) à partir du **contexte de l'activité**
- Installation du layout via `inflate` (le parent est le *Spinner*)

Application exemple : `ItemCategorySpinnerAdapter`

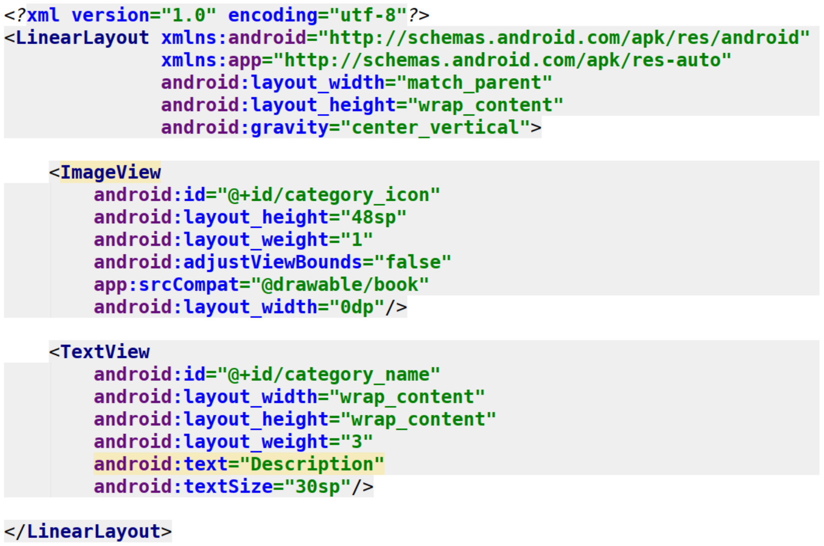


```java
public View getView(int position, View convertView, ViewGroup parent) {
    if (convertView == null) {
        convertView = LayoutInflater.from(this.context)
                .inflate(R.layout.category_layout, parent, attachToRoot: false);
    }

    ItemCategory itemCategory = (ItemCategory) this.getItem(position);

    TextView textView = convertView.findViewById(R.id.category_name);
    textView.setText(itemCategory.getDescription());

    ImageView imageView = convertView.findViewById(R.id.category_icon);

    switch (itemCategory) {...}

    return convertView;
}
```

- Obtention de l'objet correspondant à la position demandée
- Mise à jour des éléments de la vue à partir de cet objet
    - TextView

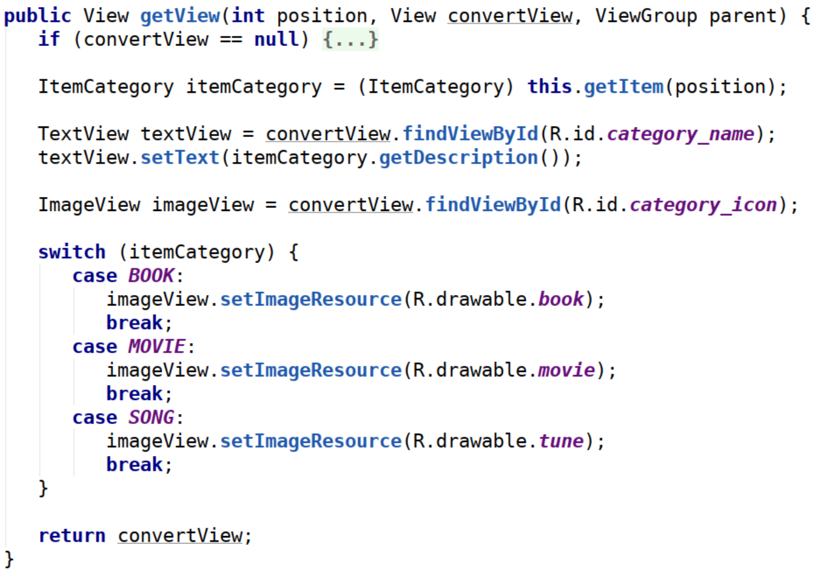
Application exemple : ItemCategorySpinnerAdapter

Layout d'un élément du spinner



```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
              xmlns:app="http://schemas.android.com/apk/res-auto"
              android:layout_width="match_parent"
              android:layout_height="wrap_content"
              android:gravity="center_vertical">

    <ImageView
        android:id="@+id/category_icon"
        android:layout_height="48sp"
        android:layout_weight="1"
        android:adjustViewBounds="false"
        app:srcCompat="@drawable/book"
        android:layout_width="0dp"/>

    <TextView
        android:id="@+id/category_name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="3"
        android:text="Description"
        android:textSize="30sp"/>

</LinearLayout>
```

Application exemple : ItemCategorySpinnerAdapter


```
public View getView(int position, View convertView, ViewGroup parent) {
    if (convertView == null) {...}

    ItemCategory itemCategory = (ItemCategory) this.getItem(position);

    TextView textView = convertView.findViewById(R.id.category_name);
    textView.setText(itemCategory.getDescription());

    ImageView imageView = convertView.findViewById(R.id.category_icon);

    switch (itemCategory) {
        case BOOK:
            imageView.setImageResource(R.drawable.book);
            break;
        case MOVIE:
            imageView.setImageResource(R.drawable.movie);
            break;
        case SONG:
            imageView.setImageResource(R.drawable.tune);
            break;
    }

    return convertView;
}
```

- Mise à jour de l'*ImageView*

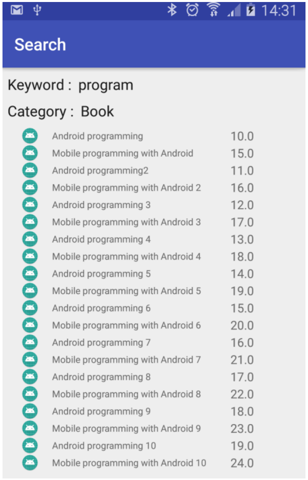
Application exemple : ItemCategorySpinnerAdapter

Autres méthodes de SpinnerAdapter

```
@Override
public boolean isEmpty() { return false; }

@Override
public int getItemViewType(int position) { return 0; }

@Override
public int getViewTypeCount() { return 1; }

@Override
public long getItemId(int position) { return position; }

@Override
public boolean hasStableIds() { return true; }

@Override
public void registerDataSetObserver(DataSetObserver dataSetObserver) {}

@Override
public void unregisterDataSetObserver(DataSetObserver dataSetObserver) {}
```

- Vigilance sur : *getViewTypeCount*

Application exemple : layout de l'activité `ResultActivity`



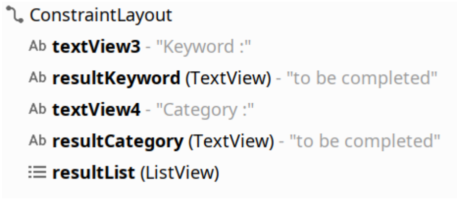
14:31

Search

Keyword : program

Category : Book

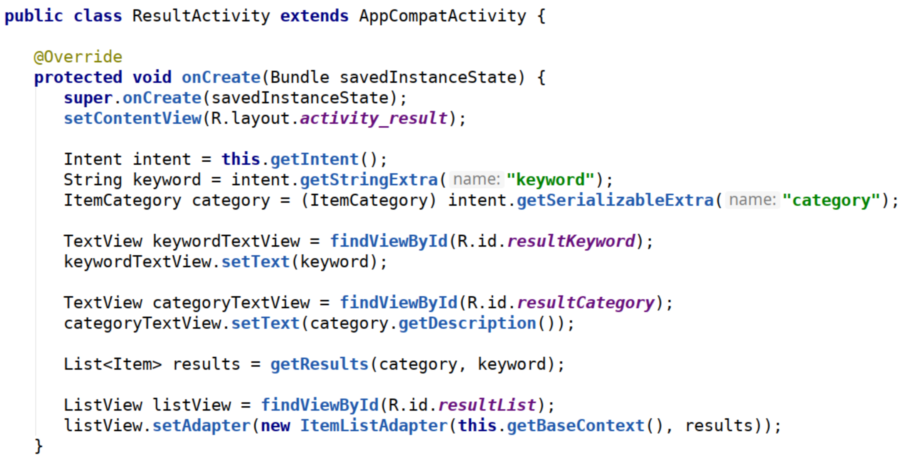
Android programming	10.0
Mobile programming with Android	15.0
Android programming2	11.0
Mobile programming with Android 2	16.0
Android programming 3	12.0
Mobile programming with Android 3	17.0
Android programming 4	13.0
Mobile programming with Android 4	18.0
Android programming 5	14.0
Mobile programming with Android 5	19.0
Android programming 6	15.0
Mobile programming with Android 6	20.0
Android programming 7	16.0
Mobile programming with Android 7	21.0
Android programming 8	17.0
Mobile programming with Android 8	22.0
Android programming 9	18.0
Mobile programming with Android 9	23.0
Android programming 10	19.0
Mobile programming with Android 10	24.0



ConstraintLayout
- Ab **textView3** - "Keyword :"
- Ab **resultKeyword** (TextView) - "to be completed"
- Ab **textView4** - "Category :"
- Ab **resultCategory** (TextView) - "to be completed"
- **resultList** (ListView)

Application exemple : Code de `ResultActivity`

onCreate

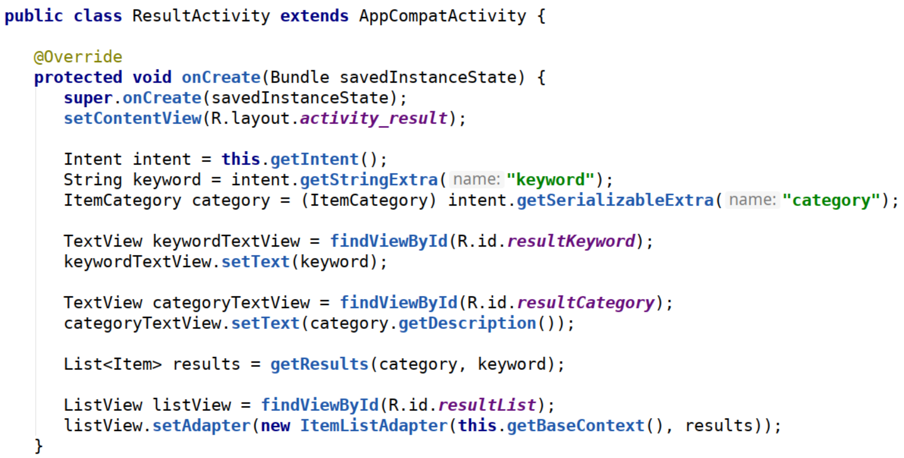


```java
public class ResultActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_result);

        Intent intent = this.getIntent();
        String keyword = intent.getStringExtra( name: "keyword");
        ItemCategory category = (ItemCategory) intent.getSerializableExtra( name: "category");

        TextView keywordTextView = findViewById(R.id.resultKeyword);
        keywordTextView.setText(keyword);

        TextView categoryTextView = findViewById(R.id.resultCategory);
        categoryTextView.setText(category.getDescription());

        List<Item> results = getResults(category, keyword);

        ListView listView = findViewById(R.id.resultList);
        listView.setAdapter(new ItemListAdapter(this.getBaseContext(), results));
    }
```

Récupération des critères de recherche dans l'intent

Application exemple : Code de `ResultActivity`

onCreate

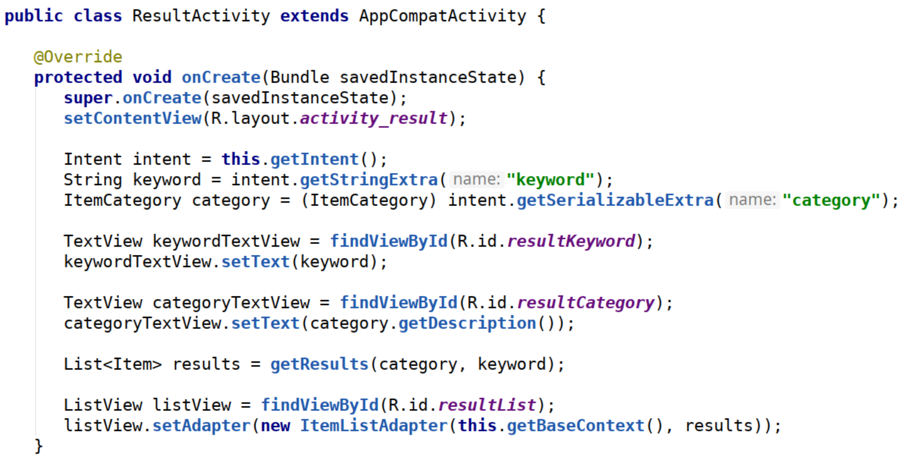


```java
public class ResultActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_result);

        Intent intent = this.getIntent();
        String keyword = intent.getStringExtra( name: "keyword");
        ItemCategory category = (ItemCategory) intent.getSerializableExtra( name: "category");

        TextView keywordTextView = findViewById(R.id.resultKeyword);
        keywordTextView.setText(keyword);

        TextView categoryTextView = findViewById(R.id.resultCategory);
        categoryTextView.setText(category.getDescription());

        List<Item> results = getResults(category, keyword);

        ListView listView = findViewById(R.id.resultList);
        listView.setAdapter(new ItemListAdapter(this.getBaseContext(), results));
    }
```

Mise à jour des champs de texte

Application exemple : Code de `ResultActivity`

onCreate

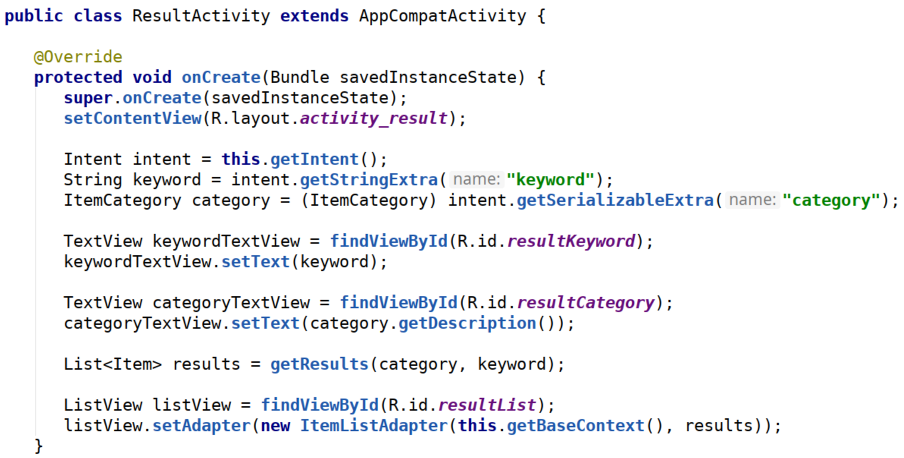


```java
public class ResultActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_result);

        Intent intent = this.getIntent();
        String keyword = intent.getStringExtra( name: "keyword");
        ItemCategory category = (ItemCategory) intent.getSerializableExtra( name: "category");

        TextView keywordTextView = findViewById(R.id.resultKeyword);
        keywordTextView.setText(keyword);

        TextView categoryTextView = findViewById(R.id.resultCategory);
        categoryTextView.setText(category.getDescription());

        List<Item> results = getResults(category, keyword);

        ListView listView = findViewById(R.id.resultList);
        listView.setAdapter(new ItemListAdapter(this.getBaseContext(), results));
    }
```

Recherche des items correspondant

Application exemple : Code de `ResultActivity`

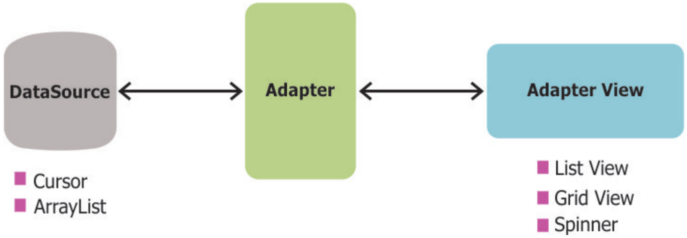
onCreate



```java
public class ResultActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_result);

        Intent intent = this.getIntent();
        String keyword = intent.getStringExtra( name: "keyword");
        ItemCategory category = (ItemCategory) intent.getSerializableExtra( name: "category");

        TextView keywordTextView = findViewById(R.id.resultKeyword);
        keywordTextView.setText(keyword);

        TextView categoryTextView = findViewById(R.id.resultCategory);
        categoryTextView.setText(category.getDescription());

        List<Item> results = getResults(category, keyword);

        ListView listView = findViewById(R.id.resultList);
        listView.setAdapter(new ItemListAdapter(this.getBaseContext(), results));
    }
```

Mise en place de l'adapter du `ListView`

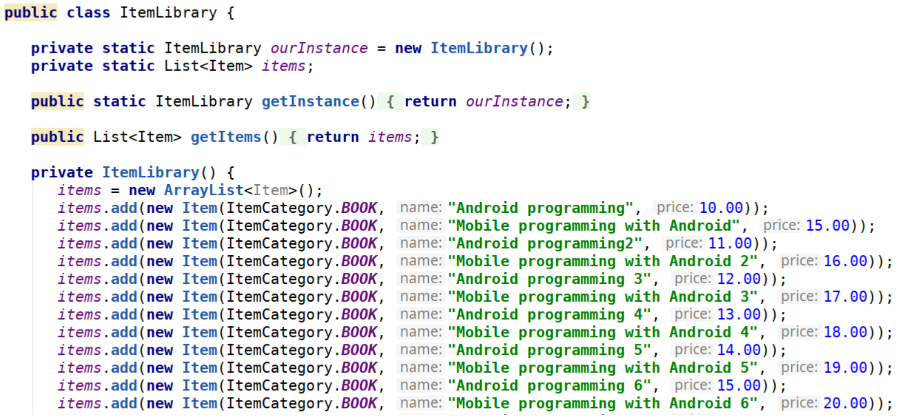
Association d'un adapter à une `GroupView`



source

Stockage des données

Utilisation d'un Singleton



```
public class ItemLibrary {

    private static ItemLibrary ourInstance = new ItemLibrary();
    private static List<Item> items;

    public static ItemLibrary getInstance() { return ourInstance; }

    public List<Item> getItems() { return items; }

    private ItemLibrary() {
        items = new ArrayList<Item>();
        items.add(new Item(ItemCategory.BOOK,  name: "Android programming",  price: 10.00));
        items.add(new Item(ItemCategory.BOOK,  name: "Mobile programming with Android",  price: 15.00));
        items.add(new Item(ItemCategory.BOOK,  name: "Android programming2",  price: 11.00));
        items.add(new Item(ItemCategory.BOOK,  name: "Mobile programming with Android 2",  price: 16.00));
        items.add(new Item(ItemCategory.BOOK,  name: "Android programming 3",  price: 12.00));
        items.add(new Item(ItemCategory.BOOK,  name: "Mobile programming with Android 3",  price: 17.00));
        items.add(new Item(ItemCategory.BOOK,  name: "Android programming 4",  price: 13.00));
        items.add(new Item(ItemCategory.BOOK,  name: "Mobile programming with Android 4",  price: 18.00));
        items.add(new Item(ItemCategory.BOOK,  name: "Android programming 5",  price: 14.00));
        items.add(new Item(ItemCategory.BOOK,  name: "Mobile programming with Android 5",  price: 19.00));
        items.add(new Item(ItemCategory.BOOK,  name: "Android programming 6",  price: 15.00));
        items.add(new Item(ItemCategory.BOOK,  name: "Mobile programming with Android 6",  price: 20.00));
```

. . .

Création de l'Adapter

Comme pour le SpinnerAdapter :

- Redéfinition des méthodes `getCount`, `getItem`, `getView`, etc.
- Stockage de la liste d'items à afficher lors de la construction
- Définition d'un `OnClickListener` qui sera associé à chaque ligne
  - Cet `OnClickListener` est une fonction de `ResultActivity`
  - Il faut un moyen simple de retrouver l'élément cliqué

    → `setTag`