

Recycler View

La Recycler View est un élément qui permet d'afficher une liste d'éléments. Contrairement à la List View, la Recycler View va effectuer tout un tas d'optimisations sous le capot afin de pouvoir afficher avec de bonnes performances un grand nombre d'éléments.

Créer le modèle

Nous allons avoir besoin d'une classe de modèle pour afficher des éléments dans notre liste, créez une simple classe `Todo` dans le package `model` :

```
public class Todo {  
    private String id;  
    private String title;  
    private String description;  
  
    ... constructeur, getters & setters ...  
}
```

Créer la Recycler View

Pour créer votre Recycler View, aller tout simplement dans le Layout de votre `TodoListFragment`, recherchez le dans la liste des contrôles et ajoutez le.

Adapteur de Liste

Nous allons avoir besoin de développer une classe Adapteur afin de faire le lien entre notre Liste d'objet `Todo`, la Recycler View et notre Layout d'Item. Créez donc une classe `TodoListAdapter` dans le package `view` qui étend la classe `RecyclerView.Adapter` :

```
public class TodoListAdapter extends RecyclerView.Adapter<> {  
  
}
```

Ensuite, nous devons créer un ViewHolder, c'est un pattern imposé par la RecyclerView mais dans notre cas il ne fera rien, donc nous n'allons pas rentrer dans les détails. Pour ce faire, créez une classe interne `TodoViewHolder` qui étend `RecyclerView.ViewHolder`, ajoutez la en paramètre de type de notre adaptateur :

```
public class TodoListAdapter extends RecyclerView.Adapter<TodoViewHolder> {

    class TodoViewHolder extends RecyclerView.ViewHolder {

        public View itemView;

        public TodoViewHolder(@NonNull View itemView) {
            super(itemView);
            this.itemView = itemView;
        }
    }
}
```

Il nous faut aussi un attributs pour stocker les éléments de notre liste, et un constructeur qui les récupère :

```
private List<Todo> items;

public TodoListAdapter(List<Todo> items) {
    this.items = items;
}
```

Et également un setter pour mettre à jours la liste :

```
public void setDogList(List<DogBreed> dogList) {
    this.dogList.clear();
    this.dogList.addAll(dogList);
    notifyDataSetChanged();
}
```

L'appelle à `notifyDataSetChanged` va notifier la RecyclerView qu'il vaut se mettre à jours.

Enfin, il faut implémenter les méthodes `onCreateViewHolder`, `onBindViewHolder` et `getItemCount`.

- `getItemCount` : Va tout simplement retourner le nombre d'Items dans la liste :

```
@Override
public int getItemCount() {
    return this.items.size();
}
```

- `onCreateViewHolder` : Va s'occuper d'inflater la `View` qui correspond à un Item à partir de notre Item Layout :

```
@NonNull
@Override
public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.todo_list_item, parent, false);
    return new ViewHolder(view);
}
```

- `onBindViewHolder` : Va s'occuper de remplir les éléments de la vue avec les informations du modèle :

```
@Override
public void onBindViewHolder(@NonNull ViewHolder holder, int position) {
    TextView title = holder.itemView.findViewById(R.id.todo_list_item_title);
    TextView description = holder.itemView.findViewById(R.id.todo_list_item_description);

    title.setText(items.get(position).getTitle());
    description.setText(items.get(position).getDescription());
}
```

Rajoutez également un setter pour la liste Todos, en cas de besoin de changer le contenu de la liste. Appelez, `notifyDataSetChanged` à la fin de ce setter afin de mettre à jours le rendu de la liste avec les nouvelles données.

Connecter RecyclerView et Adapter

Dans notre `TodoListFragment`, commençons par récupérer notre `RecyclerView` avec `ButterKnife` :

```
@BindView(R.id.recyclerView)
public RecyclerView recyclerView;
```

Dans la méthode `onViewCreated`, nous allons ensuite par créer des données de test et les connecter à notre Adapter et notre RecyclerView :

```
List<Todo> todos = Arrays.asList(
    new Todo(UUID.randomUUID().toString(), "Test Todo Title 1", "Test Todo Description 1"),
    new Todo(UUID.randomUUID().toString(), "Test Todo Title 2", "Test Todo Description 2"),
    new Todo(UUID.randomUUID().toString(), "Test Todo Title 3", "Test Todo Description 3"),
    new Todo(UUID.randomUUID().toString(), "Test Todo Title 4", "Test Todo Description 4")

    ....

);

TodoListAdapter adapter = new TodoListAdapter(todos);
recyclerView.setLayoutManager(new LinearLayoutManager(getContext()));
recyclerView.setAdapter(adapter);
```

Cela permet bien d'afficher notre Liste :

Todo

Test Todo Title 1

Test Todo Description 1

Test Todo Title 2

Test Todo Description 2

Test Todo Title 3

Test Todo Description 3

Test Todo Title 4

Test Todo Description 4

Test Todo Title 1

Test Todo Description 1

Test Todo Title 2

Revision #6

Created 7 February 2021 15:22:59 by Arsène Lapostolet

Updated 9 February 2021 18:49:36 by Arsène Lapostolet