TP06 – Utilisation d'un RecyclerView

Objectifs :

1. Comprendre et implémenter un modèle MVC avec RecyclerView.
2. Utiliser un ArrayList comme source de données dynamique.
3. Gérer les clics sur les items pour afficher des détails.
4. Permettre l'ajout et la suppression d'éléments dans la liste.

Étape 1 : Configuration Initiale

1. Créer un nouveau projet Android Studio.
   - Nom : PlanetsApp.
   - Langage : Java.
   - Activité de base : Empty Activity.

Étape 2 : Définir la Structure des Données

1. Créer une classe Planete :

```
public class Planete {
    private String nom;
    private int distance;

    public Planete(String nom, int distance) {
        this.nom = nom;
        this.distance = distance;
    }

    public String getNom() {
        return nom;
    }

    public void setNom(String nom) {
        this.nom = nom;
    }

    public int getDistance() {
        return distance;
    }

    public void setDistance(int distance) {
        this.distance = distance;
    }
}
```

2. Créer une liste initiale des planètes dans MainActivity :

```
private List<Planete> initialiserPlanetes() {
    return new ArrayList<>(Arrays.asList(
        new Planete("Mercure", 58),
        new Planete("Vénus", 108),
        new Planete("Terre", 150),
        new Planete("Mars", 228),
        new Planete("Jupiter", 778)
    ));
}
```

TP06 – Utilisation d'un RecyclerView

Étape 3 : Créer les Layouts

1. Layout principal : activity_main.xml.

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerView"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

2. Layout pour un item : item_planete.xml.

```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="16dp">

    <TextView
        android:id="@+id/nom"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="18sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/distance"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/nom"
        android:textSize="14sp" />
</RelativeLayout>
```

Étape 4 : Créer l'Adapter

1. Créer la classe PlaneteAdapter :

```
public class PlaneteAdapter extends
RecyclerView.Adapter<PlaneteAdapter.PlaneteViewHolder> {
    private final List<Planete> planetes;
    private final OnItemClickListener listener;

    public PlaneteAdapter(List<Planete> planetes, OnItemClickListener
listener) {
        this.planetes = planetes;
        this.listener = listener;
    }

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

    @Override
    public void onBindViewHolder(@NonNull PlaneteViewHolder holder, int
position) {
```

TP06 – Utilisation d'un RecyclerView

```
        Planete planete = planetes.get(position);
        holder.nom.setText(planete.getNom());
        holder.distance.setText(planete.getDistance() + " Gm");
        holder.itemView.setOnClickListener(v ->
listener.onItemClick(planete, position));
    }

    @Override
    public int getItemCount() {
        return planetes.size();
    }

    public interface OnItemClickListener {
        void onItemClick(Planete planete, int position);
    }

    static class PlaneteViewHolder extends RecyclerView.ViewHolder {
        TextView nom, distance;

        public PlaneteViewHolder(@NonNull View itemView) {
            super(itemView);
            nom = itemView.findViewById(R.id.nom);
            distance = itemView.findViewById(R.id.distance);
        }
    }
}
```

Étape 5 : Configurer le RecyclerView

1. Configurer dans MainActivity :

```
private RecyclerView recyclerView;
private PlaneteAdapter adapter;
private List<Planete> listePlanetes;

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

    recyclerView = findViewById(R.id.recyclerView);
    listePlanetes = initialiserPlanetes();

    adapter = new PlaneteAdapter(listePlanetes, (planete, position) -> {
        Toast.makeText(this, "Clique sur : " + planete.getNom(),
Toast.LENGTH_SHORT).show();
    });
    recyclerView.setLayoutManager(new LinearLayoutManager(this));
    recyclerView.setAdapter(adapter);
}
```

Étape 6 : Ajouter des Fonctionnalités

1. Ajout d'une planète avec un bouton :
   - Ajouter un bouton dans activity_main.xml.

```
<Button
    android:id="@+id/btnAjouter"
    android:layout_width="wrap_content"
```

TP06 – Utilisation d'un RecyclerView

```
        android:layout_height="wrap_content"
        android:text="Ajouter Planète"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true" />
```

- Gérer l'ajout dans MainActivity :

```
Button btnAjouter = findViewById(R.id.btnAjouter);
btnAjouter.setOnClickListener(v -> {
    Planete nouvellePlanete = new Planete("Saturne", 1427);
    listePlanetes.add(nouvellePlanete);
    adapter.notifyItemInserted(listePlanetes.size() - 1);
});
```

2. Ajouter une Interface pour Gérer les Clics Longs
   - Déclarez une interface dans la classe PlaneteAdapter pour notifier la position de l'élément cliqué.

```
public interface OnItemLongClickListener {
    void onItemLongClick(int position);
}
```

3. Ajouter un Champ pour l'Écouteur dans l'Adapter
   - Ajoutez une variable pour stocker une instance de l'écouteur et un setter.

```
private OnItemLongClickListener longClickListener;

public void setOnItemLongClickListener(OnItemLongClickListener listener) {
    this.longClickListener = listener;
}
```

4. Implémenter le Clic Long dans le ViewHolder
   - Ajoutez un écouteur de clic long dans onBindViewHolder :

```
@Override
public void onBindViewHolder(@NonNull PlaneteViewHolder holder, int
position) {
    Planete planete = planetes.get(position);
    holder.nom.setText(planete.getNom());
    holder.distance.setText(planete.getDistance() + " Gm");
    holder.image.setImageResource(planete.getImageResId());

    holder.itemView.setOnLongClickListener(v -> {
        if (longClickListener != null) {
            longClickListener.onItemLongClick(position);
        }
        return true;
    });
}
```

5. Configurer l'Écouteur dans MainActivity
   - Définissez ce qui doit se produire lors du clic long sur un item :

```
adapter.setOnItemLongClickListener(position -> {

    listePlanetes.remove(position);
    adapter.notifyItemRemoved(position);
    Toast.makeText(MainActivity.this, "Planète supprimée",
Toast.LENGTH_SHORT).show();
});
```

TP06 – Utilisation d'un RecyclerView

6. Confirmation de la Suppression
   - Affichez une boîte de dialogue pour demander à l'utilisateur de confirmer avant de supprimer la planète.

```
new AlertDialog.Builder(MainActivity.this)
    .setTitle("Confirmation")
    .setMessage("Voulez-vous vraiment supprimer cette planète ?")
    .setPositiveButton("Oui", (dialog, which) -> {
        listePlanetes.remove(position);
        adapter.notifyItemRemoved(position);
    })
    .setNegativeButton("Non", null)
    .show();
```

Étape 7 : Ajout d'Images aux Planètes

7. Modifier la Classe Planete :
   - Ajoutez un champ pour représenter l'image :

```
public class Planete {
    private String nom;
    private int distance;
    private int imageResId; // ID de la ressource image

    public Planete(String nom, int distance, int imageResId) {
        this.nom = nom;
        this.distance = distance;
        this.imageResId = imageResId;
    }

    // Getters et Setters pour l'image
    public int getImageResId() {
        return imageResId;
    }

    public void setImageResId(int imageResId) {
        this.imageResId = imageResId;
    }
}
```

8. Modifier le Layout des Items (item_planete.xml) :
   - Ajoutez une ImageView :

```
<ImageView
    android:id="@+id/image"
    android:layout_width="64dp"
    android:layout_height="64dp"
    android:scaleType="centerCrop"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintBottom_toBottomOf="parent" />
```

9. Mettre à Jour l'Adapter :
   - Chargez les images dans la méthode onBindViewHolder :

```
@Override
public void onBindViewHolder(@NonNull PlaneteViewHolder holder, int
position) {
    Planete planete = planetes.get(position);
```

TP06 – Utilisation d'un RecyclerView

```
        holder.nom.setText(planete.getNom());
        holder.distance.setText(planete.getDistance() + " Gm");
        holder.image.setImageResource(planete.getImageResId());
    }
```

10. Ajouter des Images aux Données Initiales :
    - Mettez à jour les données des planètes avec des IDs d'images :

```
private List<Planete> initialiserPlanetes() {
    return new ArrayList<>(Arrays.asList(
        new Planete("Mercure", 58, R.drawable.mercury),
        new Planete("Vénus", 108, R.drawable.venus),
        new Planete("Terre", 150, R.drawable.earth),
        new Planete("Mars", 228, R.drawable.mars)
    ));
}
```

Étape 8 : Ajouter un Menu Contextuel

11. Créer un Menu Contextuel (menu_contextuel.xml) :
    - Créez un fichier XML dans res/menu :

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/action_modify"
        android:title="Modifier" />
    <item
        android:id="@+id/action_delete"
        android:title="Supprimer" />
</menu>
```

12. Modifier l'Adapter pour Gérer les Clics Longs :
    - Implémentez le clic long sur les items :

```
@Override
public void onBindViewHolder(@NonNull PlaneteViewHolder holder, int
position) {
    Planete planete = planetes.get(position);
    holder.nom.setText(planete.getNom());
    holder.distance.setText(planete.getDistance() + " Gm");
    holder.image.setImageResource(planete.getImageResId());

    holder.itemView.setOnLongClickListener(v -> {
        listener.onItemLongClick(planete, position, v);
        return true;
    });
}

public interface OnItemLongClickListener {
    void onItemLongClick(Planete planete, int position, View view);
}
```

13. Configurer le Menu Contextuel dans MainActivity :
    - Gérer l'affichage et les actions du menu :

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v,
ContextMenu.ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);
    getMenuInflater().inflate(R.menu.menu_contextuel, menu);
```

TP06 – Utilisation d'un RecyclerView

```
}

@Override
public boolean onContextItemSelected(@NonNull MenuItem item) {
    int position = adapter.getSelectedItemPosition(); // Position
récupérée
    switch (item.getItemId()) {
        case R.id.action_modify:
            // Logique de modification
            Toast.makeText(this, "Modifier : " +
listePlanetes.get(position).getNom(), Toast.LENGTH_SHORT).show();
            return true;
        case R.id.action_delete:
            listePlanetes.remove(position);
            adapter.notifyItemRemoved(position);
            Toast.makeText(this, "Planète supprimée",
Toast.LENGTH_SHORT).show();
            return true;
        default:
            return super.onContextItemSelected(item);
    }
}
```

14. Ajouter une Méthode pour Récupérer la Position Sélectionnée dans l'Adapter :

```
private int selectedPosition;

public int getSelectedItemPosition() {
    return selectedPosition;
}

public void setSelectedItemPosition(int position) {
    this.selectedPosition = position;
}
```