**RecyclerView** Drag&Drop , Swipe-to-dismiss

**RecyclerView** Drag&Drop , Swipe-to-dismiss

**`ItemTouchHelper`**

Es una librería que se encarga de proporcionar todo lo necesario para agregar la capacidad de realizar Drag&Drop y Swipe-to-dismiss entre los items de un RecyclerView

**RecyclerView** Drag&Drop , Swipe-to-dismiss

## `ItemTouchHelper` 0 1 2

Para crear un objeto ItemTouchHelper es necesario crear previamente un objeto **`ItemTouchHelper.Callback`**

# RecyclerView

## Drag&Drop , Swipe-to-dismiss

### ItemTouchHelper 0 1 2

### ItemTouchHelper.Callback 0 1 2 3 4 5

```
public static class CountriesItemTouchHelperCallback extends ItemTouchHelper.Callback
{
1 @Override public boolean isLongPressDragEnabled() { … }

2 @Override public boolean isItemViewSwipeEnabled() { … }

3 @Override public int getMovementFlags(RecyclerView recyclerView, RecyclerView.ViewHolder viewHolder) {…}

4 @Override
  public boolean onMove(RecyclerView recyclerView, RecyclerView.ViewHolder viewHolder, RecyclerView.ViewHolder target) {…}

5 @Override public void onSwiped(RecyclerView.ViewHolder viewHolder, int direction) { … }

}
```

¿Se desea realizar Drag&Drop?

¿Se desea realizar Swipe?

¿Cómo se activa Drag&Drop y Swipe?

¿Qué ejecutar si Drag&Drop?

¿Qué ejecutar si Swipe-to-dismiss?

# RecyclerView

## Drag&Drop , Swipe-to-dismiss

### ItemTouchHelper 0 1 2

### ItemTouchHelper.Callback 0 1 2 3 4 5

```
public static class CountriesItemTouchHelperCallback extends ItemTouchHelper.Callback
{


  @Override
  public boolean isLongPressDragEnabled()
  {
    return true;
  }
```

¿Se desea realizar Drag&Drop?

# RecyclerView Drag&Drop , Swipe-to-dismiss

## ItemTouchHelper 0 1 2

### ItemTouchHelper.Callback 0 1 2 3 4 5

```
public static class CountriesItemTouchHelperCallback extends ItemTouchHelper.Callback
{


  @Override
  public boolean isItemViewSwipeEnabled()
  {
    return true;
  }
```

¿Se desea realizar Swipe?

# RecyclerView

## Drag&Drop , Swipe-to-dismiss

### ItemTouchHelper 0 1 2

### ItemTouchHelper.Callback 0 1 2 3 4 5

```
public static class CountriesItemTouchHelperCallback extends ItemTouchHelper.Callback
{
```

¿Cómo se activa Drag&Drop y Swipe?

```
  @Override
  public int getMovementFlags(RecyclerView recyclerView, RecyclerView.ViewHolder viewHolder)
  {
    int dragFlags  = ItemTouchHelper.UP   | ItemTouchHelper.DOWN;
    int swipeFlags = ItemTouchHelper.LEFT | ItemTouchHelper.RIGHT;

    return ItemTouchHelper.Callback.makeMovementFlags(dragFlags, swipeFlags);
  }
```

# RecyclerView Drag&Drop , Swipe-to-dismiss

## ItemTouchHelper 0 1 2

### ItemTouchHelper.Callback 0 1 2 3 4 5

```
public static class CountriesItemTouchHelperCallback extends ItemTouchHelper.Callback
{

  @Override
  public boolean onMove(RecyclerView recyclerView,
                        RecyclerView.ViewHolder viewHolder,
                        RecyclerView.ViewHolder target)
  {
    int fromPosition = viewHolder.getAdapterPosition();
    int toPosition   = target.getAdapterPosition();

    // Mover el elemento del RecyclerView con posición fromPosition a la posición toPosition
    return true;
  }
```

¿Qué ejecutar si Drag&Drop?

Habría que definir un método en el *adapter* del RecyclerView que además de actualizar los datos subyacentes, también notificase del movimiento al *adapter* a fin de actualizar las vistas de los items afectados del RecyclerView. La mejor forma de hacerlo sería:

a) definir un interface.
b) implementar el interface en el *adapter*
c) pasar la referencia del interface como parámetro en el constructor de ItemTouchHelper.Callback

# RecyclerView

## Drag&Drop , Swipe-to-dismiss

### ItemTouchHelper 0 1 2

### ItemTouchHelper.Callback 0 1 2 3 4 5

```
public static class CountriesItemTouchHelperCallback extends ItemTouchHelper.Callback
{


  @Override
  public void onSwipe(RecyclerView.ViewHolder viewHolder,
                      int direction)
  {
    int position = viewHolder.getAdapterPosition();

    // Eliminar el elemento del RecyclerView
    // con posición position
  }
```

¿Qué ejecutar si Swipe-to-dismiss?

Habría que definir un método en el *adapter* del RecyclerView que además de actualizar los datos subyacentes, también notificase del movimiento al *adapter* a fin de actualizar las vistas de los items afectados del RecyclerView. La mejor forma de hacerlo sería:

a) definir un interface.
b) implementar el interface en el *adapter*
c) pasar la referencia del interface como parámetro en el constructor de ItemTouchHelper.Callback

# RecyclerView

## Drag&Drop , Swipe-to-dismiss

### ItemTouchHelper 0 1 2

### ItemTouchHelper.Callback 0 1 2 3 4 4.a 5 5.a

```
public interface OnMoveAndDismissListener
{
 4.a public void onMove(int fromPosition, int toPosition);

 5.a public void onDismiss(int position);

}
```

Definición del interface a través del que definir funcionalidades con las que mover y eliminar elementos

# RecyclerView Drag&Drop , Swipe-to-dismiss

## ItemTouchHelper 0 1 2

## ItemTouchHelper.Callback 0 1 2 3 4 4.b 5 5.b

```
public class CountriesAdapter extends    RecyclerView.Adapter< CountriesAdapter.CountryViewHolder >
                              implements OnMoveAndDismissListener
{
  private List<Country> countries;

  @Override
  public void onMove(int from, int to);
  {
      if ( from < to ) for(int index= from; index < to;   ++index) Collections.swap(countries, index, index + 1);
4.b   else             for(int index= to;   index > from; --index) Collections.swap(countries, index, index - 1);
      this.notifyItemMoved(from, to);
  }

  @Override
  public void onDismiss(int position)
  {
      this.countries.remove(position);
5.b   this.notifyItemRemoved(position);
  }
```

Implementación del interface en el *adapter* del RecyclerView

# RecyclerView Drag&Drop , Swipe-to-dismiss

## ItemTouchHelper 0 1 2

## ItemTouchHelper.Callback 0 1 2 3 4 4.c 5

```
public static class CountriesItemTouchHelperCallback extends ItemTouchHelper.Callback
{
  private OnMoveAndDismissListener onMoveAndDismissListener;

  public CountriesItemTouchHelperCallback(OnMoveAndDismissListener onMoveAndDismissListener )
  {
4.c   this.onMoveAndDismissListener = onMoveAndDismissListener;
  }

  @Override
  public boolean onMove(RecyclerView recyclerView, RecyclerView.ViewHolder viewHolder, RecyclerView.ViewHolder target)
  {
      int fromPosition = viewHolder.getAdapterPosition();
4     int toPosition   = target.getAdapterPosition();

      onMoveAndDismissListener.onMove(fromPosition, toPosition);
      return true;
  }
```

Paso del interface al constructor del ItemTouchHelper.Callback y finalización de onMove()

# RecyclerView Drag&Drop , Swipe-to-dismiss

## ItemTouchHelper 0 1 2

### ItemTouchHelper.Callback 0 1 2 3 4 5 5.c

```
public static class CountriesItemTouchHelperCallback extends ItemTouchHelper.Callback
{
  private OnMoveAndDismissListener onMoveAndDismissListener;

  public CountriesItemTouchHelperCallback(OnMoveAndDismissListener onMoveAndDismissListener )
  {
  5.c this.onMoveAndDismissListener = onMoveAndDismissListener;
  }

  @Override
  public void onSwipe(RecyclerView.ViewHolder viewHolder, int direction)
  {
  5   int position = viewHolder.getAdapterPosition();
      onMoveAndDismissListener.onDismiss( position );
  }
```

Paso del interface al constructor del ItemTouchHelper.Callback y finalización de onSwipe()

# RecyclerView Drag&Drop , Swipe-to-dismiss

## ItemTouchHelper 0 1 2

```
public class MainActivity extends Activity
{
  private List<Country> countries;
  private CountriesAdapter countriesAdapter;

  private ItemTouchHelper itemTouchHelper;
  private ItemTouchHelper.Callback itemTouchHelperCallback;

  @Override
  protected void onCreate(Bundle savedInstanceState)
  {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    : : :
 a  this.countriesAdapter = new CountriesAdapter(countries);
    this.rvCountries.setAdapter(countriesAdapter);

 b  this.itemTouchHelperCallback = new CountriesItemTouchHelperCallback(countriesAdapter);
 c  itemTouchHelper = new ItemTouchHelper( itemTouchHelperCallback );
 d  itemTouchHelper.attachToRecyclerView( rvCountries );
  }

  public static class CountriesItemTouchHelperCallback extends ItemTouchHelper.Callback { … }
```

a) Creación del adapter
b) Creación del ItemTouchHelper.Callback
c) Creación del ItemTouchHelper
d) Asociación del ItemTouchHelper y el RecyclerView