

Elements Avançats de l'IU RecyclerView i CardView

Programació Multimèdia i Dispositius Mòbils



2n. DAM



RecyclerView i Cards

- Què és? i Més Info...
- Elements d'un RecyclerView
- LayoutManagers
- Com s'implementa?
- El Resultat
- Afegint Accions a la llista
- CardView





RecyclerView

<https://developer.android.com/training/material/lists-cards.html>

- **Què és?**
 - És una versió més flexible i avançada del ListView.
 - Ens permet utilitzar **d'una manera més òptima** llistes amb gran quantitat d'elements, ja que només carrega en memòria aquells ítems que l'scroll ens mostra cada vegada en pantalla, permetent-nos 'reciclar' els elements de la llista mostrats.
 - També ens permet **animar l'inserció, l'eliminació i el desplaçament** d'elements de la llista en temps real.





RecyclerView

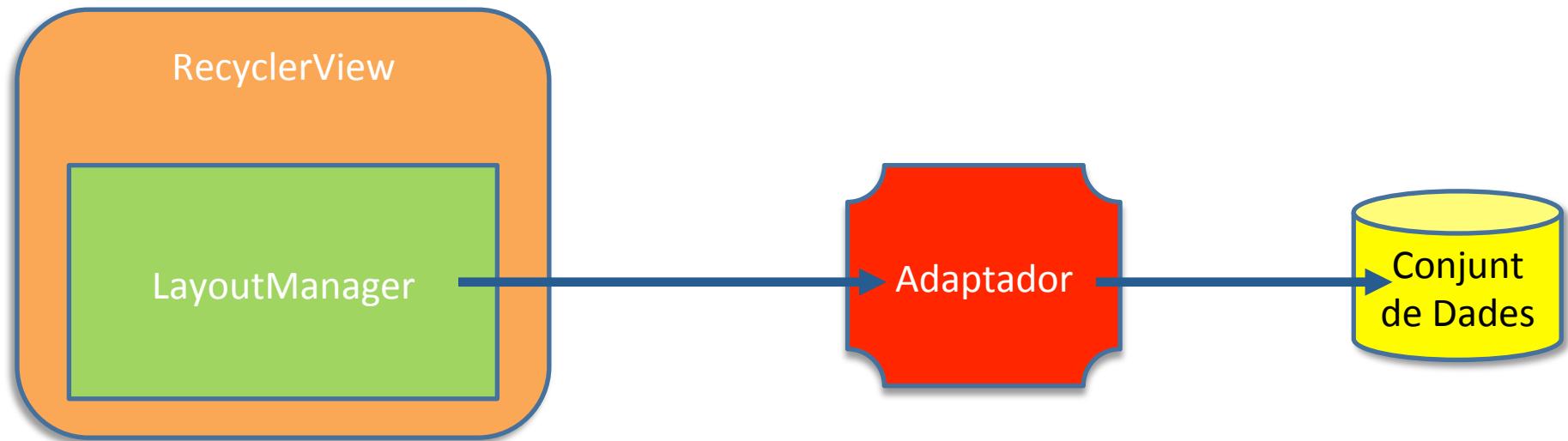
- **Més info...**

- Permeten crear scroll tant horitzontal com vertical.
- Es varen introduïr amb la versió 5 (lollipop).
- Permeten reduir **considerablement** les cridades a *findViewById(int)*, que consumeixen temps i recursos, als terminals.
- Destinats a deixar d'utilitzar definitivament ListView.





Elements d'un RecyclerView





Elements d'un RecyclerView

La creació d'un RecyclerView és similar a la del ListView.

Conjunt de dades: Ens porporciona les dades a mostrar al RecyclerView

Adaptador: Element que llig les dades, les interpreta i unfla el layout on es mostraran.

LayoutManager: S'encarrega d'afegir i reciclar els elements de la llista. Cada vegada que fem scroll sobre la llista, determina quin element ha d'eixir i quin ha d'entrar, demanant-li a l'adaptador les dades de l'element que entra. D'aquesta manera 'reciclem' el contingut d'un ítem, reduint les vegades que unflem l'element de la llista.

Amb un ListView de 100 elements, unflariem 100 vegades el layout que mostrarà l'element. Amb un RecycleView, només unflem tants layouts com elements caben en la pantalla. Millorant així l'execució i evitant la creació d'ítems que no van a ser visibles.



LayoutManagers

- RecyclerView ens proporciona 3 LayoutManagers:
LinearLayoutManager: Llista scrollable vertical o horitzontal



GridLayoutManager:
una quadrícula



Mostra els ítems en

StaggeredGridLayoutManager: Mostra els
ítems en una quadrícula escalonada





Com s'implementa?

Requeriments:

- Biblioteca de suport **v7** d'Android

Obrirem l'script build.gradle del nostre Mòdul i afegirem a l'apartat de dependències la següent línia:

```
dependencies {  
    ...  
    compile 'com.android.support:recyclerview-v7:21.0.+'  
}
```

O podeu afegir la que tingueu instal·lada.

En GitHub, tenim un exemple:

<https://github.com/mvielcor/ExempleRecyclerView.git>





Com s'implementa?

PASSOS

1. Definir el Layout per a un element de la Llista
2. Definir el Layout de l'Activity/Fragment

VISTA

3. Definir el Model de dades que utilitzarà el conjunt de dades.
4. Crear l'Adaptador per al RecyclerView

MODEL

5. Crear les dades i els objectes a l'Activity,
i enllaçar-ho tot

CONTROLADOR



1.- Definir el Layout per a un element

De la mateixa manera que feiem amb el ListView, indicarem mitjançant un XML, la distribució dels distints elements que contindrà **un únic ítem de la llista.**

Viel, Manel	123456789
PMDM	
Viel, Manel	123456789
PMDM	
Viel, Manel	123456789
PMDM	

A l'exemple, aquest fitxer conté 3 TextView on mostrar el

- cognoms, nom
- àlies
- telèfon

Organitzats convenientment.



layout_recyclerview_client.xml





2.- Definir el Layout de l'Activity

A l'Activity/fragment on anem a implementar el recyclerView, dissenyarem primer el layout de l'Activity.

En este cas l'anomenarem:

elMeuRecyclerView

Exemple de RecyclerView
Item 0
Item 1
Item 2
Item 3
Item 4
Item 5
Item 6
Item 7
Item 8
Item 9

<TextView

```
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Exemple de RecyclerView"  
    android:id="@+id/textView" />
```

<**android.support.v7.widget.RecyclerView**
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 android:id="@+id/elMeuRecyclerView"
 android:layout_alignParentStart="true"
 android:layout_below="@+id/textView"
 android:layout_alignParentEnd="true"
 android:scrollbars="vertical"/>

...





3.- Definir el Model de dades

```
public class Client {  
    private String nom;  
    private String cognom;  
    private String malnom;  
    private String telefon;  
  
    public Client(String nom, String cognom, String malnom, String telefon) {  
        this.nom = nom;  
        this.cognom = cognom;  
        this.malnom = malnom;  
        this.telefon = telefon;  
    }  
    //getters i setters  
    ...  
}
```





3.- Crear l'Adaptador

La classe Adaptador, necessita crear-se una classe interna (Holder) on definir els objectes que apareixen a l'XML d'un únic item:

```
public class AdaptadorRecyclerViewClient extends  
RecyclerView.Adapter<AdaptadorRecyclerViewClient.ClientViewHolder> {  
  
    private List<Client> llistaClients; // Llista amb els clients que volem mostrar  
  
    //Classe interna que defineix el viewHolder.  
    //Serà un objecte que ens permetrà accedir a tots els camps de l'XML que dissenya el contingut d'un item de la llista  
    public static class ClientViewHolder extends RecyclerView.ViewHolder{  
        //elements d'un item a mostrar  
        public TextView nom_i_cognoms;  
        public TextView malnom;  
        public TextView telefon;  
  
        //Constructor de ClientViewHolder  
        public ClientViewHolder(View v) {  
            super(v);  
            nom_i_cognoms = (TextView) v.findViewById(R.id.cognoms_i_nom);  
            malnom = (TextView) v.findViewById(R.id.malnom);  
            telefon = (TextView) v.findViewById(R.id.telefon);  
        }  
        ...  
    }  
}
```



3.- Crear l'Adaptador (II)

La classe adaptador tindrà un constructor que rebrà com a paràmetre les dades a mostrar.

L'adaptador, ens obliga a implementar els següents mètodes:

public int getItemCount(): Ens indica el numero d'elements de les dades a mostrar.

public ClientViewHolder onCreateViewHolder(ViewGroup parent, int viewType): Ací inflam el layout d'un item de la llista i tornem un objecte ViewHolder amb els seus atributs referencian el layout inflat.

public void onBindViewHolder(ClientViewHolder holder, int position): Ací enllacem les dades a mostrar amb un element viewHolder.





3.- Crear l'Adaptador (III)

```
@Override  
public int getItemCount(){  
    return this.llistaClients.size();  
}  
  
@Override  
public ClientViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {  
    View v = LayoutInflater.from(parent.getContext())  
        .inflate(R.layout.layout_recyclerview_client, parent, false);  
    return new ClientViewHolder(v);  
}  
  
@Override  
public void onBindViewHolder(ClientViewHolder holder, int position) {  
    holder.telefon.setText(llistaClients.get(position).getTelefon());  
    holder.nom_i_cognoms.setText(this.llistaClients.get(position).getCognoms() + ", " +  
                                this.llistaClients.get(position).getNom());  
    holder.malnom.setText(this.llistaClients.get(position).getMalnom());  
}
```





5.- Ho enllacem tot

A l'Activity principal, al mètode onCreate... ho enllacem tot

//Creem un ArrayList amb 15 clients

```
clients = new ArrayList();
```

// I l'omplim amb 15 dades de clients

```
for (int i=0;i<5;i++){  
    clients.add(new Client("Manel","Viel","PMDM","123456789"));  
    clients.add(new Client("Sr. ","Cuesta","PSP","111222333"));  
    clients.add(new Client("Baptista","Basset","PMDM","444555666"));  
}
```

//Obtenim una instància del RecyclerView

```
rv= (RecyclerView)findViewById(R.id.elMeuRecyclerView);
```

//Triem el LayoutManager que volem utilitzar i l'assignem a l'objecte recyclerView

```
rvLM = new GridLayoutManager(this,2);  
rv.setLayoutManager(rvLM);
```

//Creem l'adaptador que interactuarà amb les dades

```
aRVclient = new AdaptadorRecyclerViewClient(clients);
```

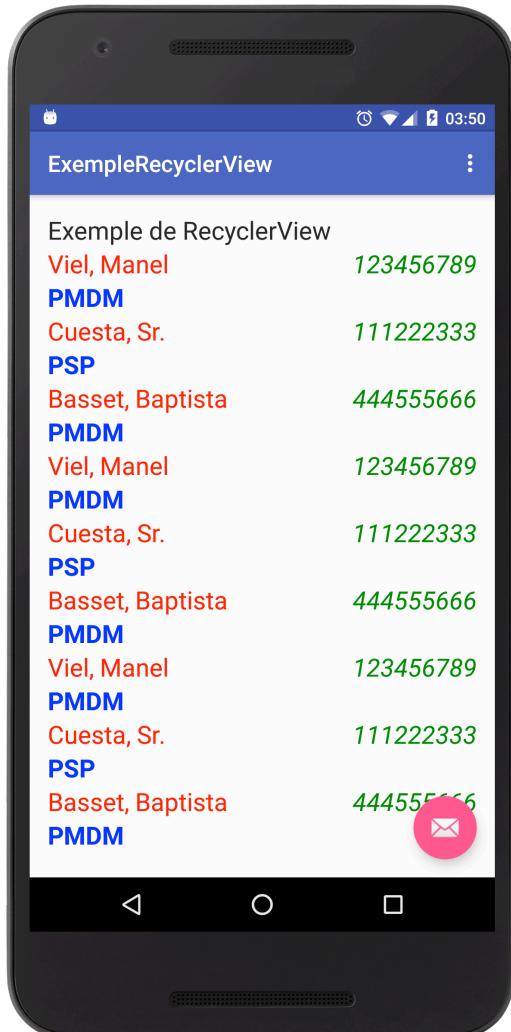
//Enllaçem el RecyclerView amb l'adaptador

```
rv.setAdapter(aRVclient);
```



El Resultat

Amb un LinearLayoutManager



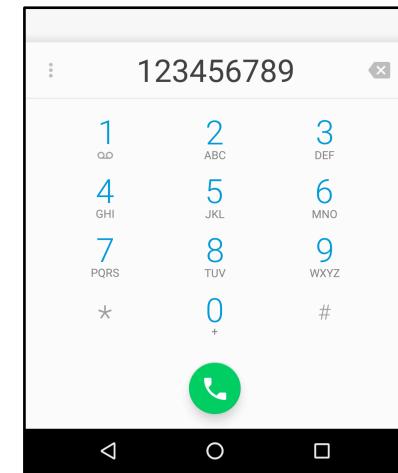
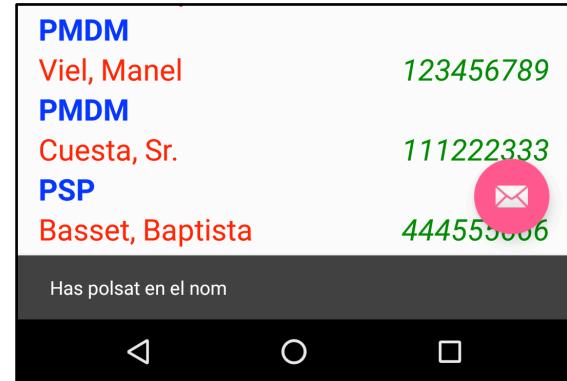
Amb un GridLayoutManager





Afegint accions a la llista

- Anem a afegir-li accions a cada element de la llista.
- Per exemple:
 - si l'usuari clica en el nom, que ens aparega un missatge indicant que ha clicat al nom
 - Si l'usuari clica en el telèfon, que ens marque el numero seleccionat per telefonar-lo.





Afegint accions a la llista

- Només cal anar al Holder (la classe interna definida a l'Adaptador), i definir els events que volem capturar:
- En principi afegirem l'OnClickListener al textview nom_i_cognoms i al textview telefon
- Després li definirem les accions a realitzar





Afegint accions a la llista

- ```
public static class ClientViewHolder extends RecyclerView.ViewHolder implements AdapterView.OnClickListener{

 //elements d'un ítem a mostrar
 public TextView nom_i_cognoms;
 public TextView malnom;
 public TextView telefon;

 //Constructor de ClientViewHolder
 public ClientViewHolder(View v) {
 super(v);
 nom_i_cognoms = (TextView) v.findViewById(R.id.cognoms_i_nom);
 nom_i_cognoms.setOnClickListener(this);
 malnom = (TextView) v.findViewById(R.id.malnom);
 telefon = (TextView) v.findViewById(R.id.telefon);
 telefon.setOnClickListener(this);
 }

 @Override
 public void onClick(View view) {
 int idView = view.getId();

 if (nom_i_cognoms.getId()==idView){ //Han fet click sobre el textView Nom_i_cognoms
 Snackbar.make(view,"Has polsat en el nom",Snackbar.LENGTH_LONG).show();
 }
 if (telefon.getId()==idView){ //Han fet click sobre el textView telefon
 TextView tv= (TextView) view;
 Intent i = new Intent(Intent.ACTION_DIAL, Uri.parse("tel:" + tv.getText().toString()));
 view.getContext().startActivity(i);
 }
 }
}
```





# CardView

Ens permeten mostrar informació dins una tarjeta amb una apariència consistent a la plataforma Android.

Ténen els cantons arrodonits, i projecten una xicoteta ombra.

Les podem utilitzar juntament amb el  
RecyclerView

Els CardViews, utilitzen elevació real iombres

Dinàmiques, gràcies al Material Design

afegit a partir de la versió 5.0

Afegirem la dependència:

`dependencies {`

`....`

`compile 'com.android.support:cardview-v7:21.0.+'`

`}`





# CardView

Per a personalitzar l'apariència de la nostra CardView, utilitzarem els següents atributs a l'XML on definim l'aspecte de cada item de la llista (a l'exemple que venim següint, al fitxer *layout\_recyclerview\_client.xml*)

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView
 xmlns:card_view="http://schemas.android.com/apk/res-auto"
 xmlns:android="http://schemas.android.com/apk/res/android"
 android:id="@+id/card_view"
 android:layout_width="match_parent"
 android:layout_height="wrap_content"
 android:layout_gravity="center"
 android:layout_margin="2dp"
 card_view:cardCornerRadius="4dp"
 card_view:cardUseCompatPadding="true"
 card_view:cardElevation="4dp">


```





# CardView

*layout\_recyclerview\_client.xml*

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView
 xmlns:card_view="http://schemas.android.com/apk/res-auto"
 xmlns:android="http://schemas.android.com/apk/res/android"
 android:id="@+id/card_view"
 android:layout_width="match_parent"
 android:layout_height="wrap_content"
 android:layout_gravity="center"
 android:layout_margin="2dp"
 card_view:cardCornerRadius="4dp"
 card_view:cardUseCompatPadding="true"
 card_view:cardElevation="4dp">
```

Esquema per a llegir els atributs del CardView



Ràdi dels cantons de la targeta, per fer-los redons

Per separar cadascuna de les targetes, en versions posteriors a la 5.0 canvien alguns atributs, mireu la doc.

Elevació de la tarjeta, per afegir-li una ombra





# CardView. El Resultat

ExempleRecyclerView :

Exemple de RecyclerView

|                  |           |
|------------------|-----------|
| Viel, Manel      | 123456789 |
| PMDM             |           |
| Cuesta, Sr.      | 111222333 |
| PSP              |           |
| Basset, Baptista | 444555666 |
| PMDM             |           |
| Viel, Manel      | 123456789 |
| PMDM             |           |
| Cuesta, Sr.      | 111222333 |
| PSP              |           |
| Basset, Baptista | 444555666 |
| PMDM             |           |
| Viel, Manel      | 123456789 |
| PMDM             |           |

A pink circular button with a white envelope icon is overlaid on the bottom right corner of the last card.