1.Recycle view

<https://www.youtube.com/watch?v=Wq2o4EbM74k>

 a)

more advanced and flexible version of [ListView](https://developer.android.com/reference/android/widget/ListView.html).

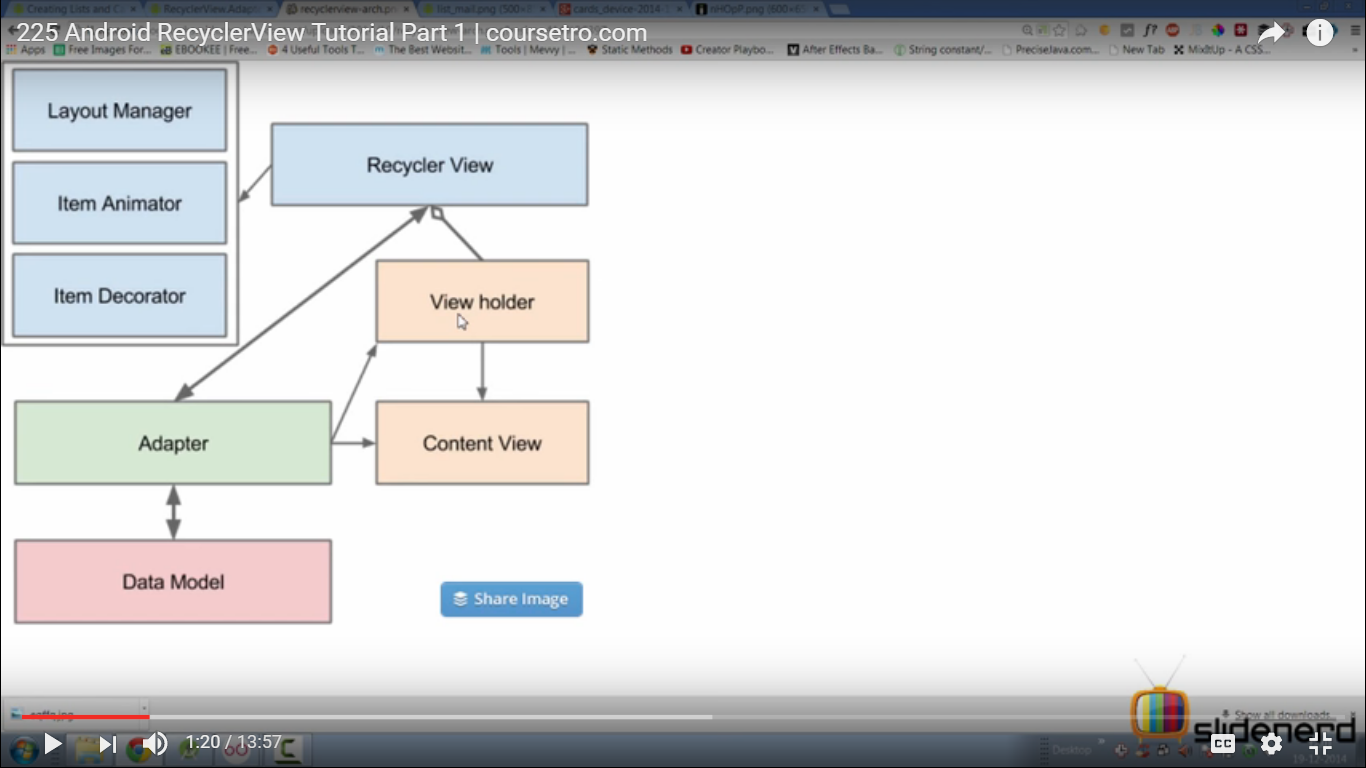
This widget is a container for displaying large data sets that can be scrolled very efficiently by maintaining a limited number of views.

b)

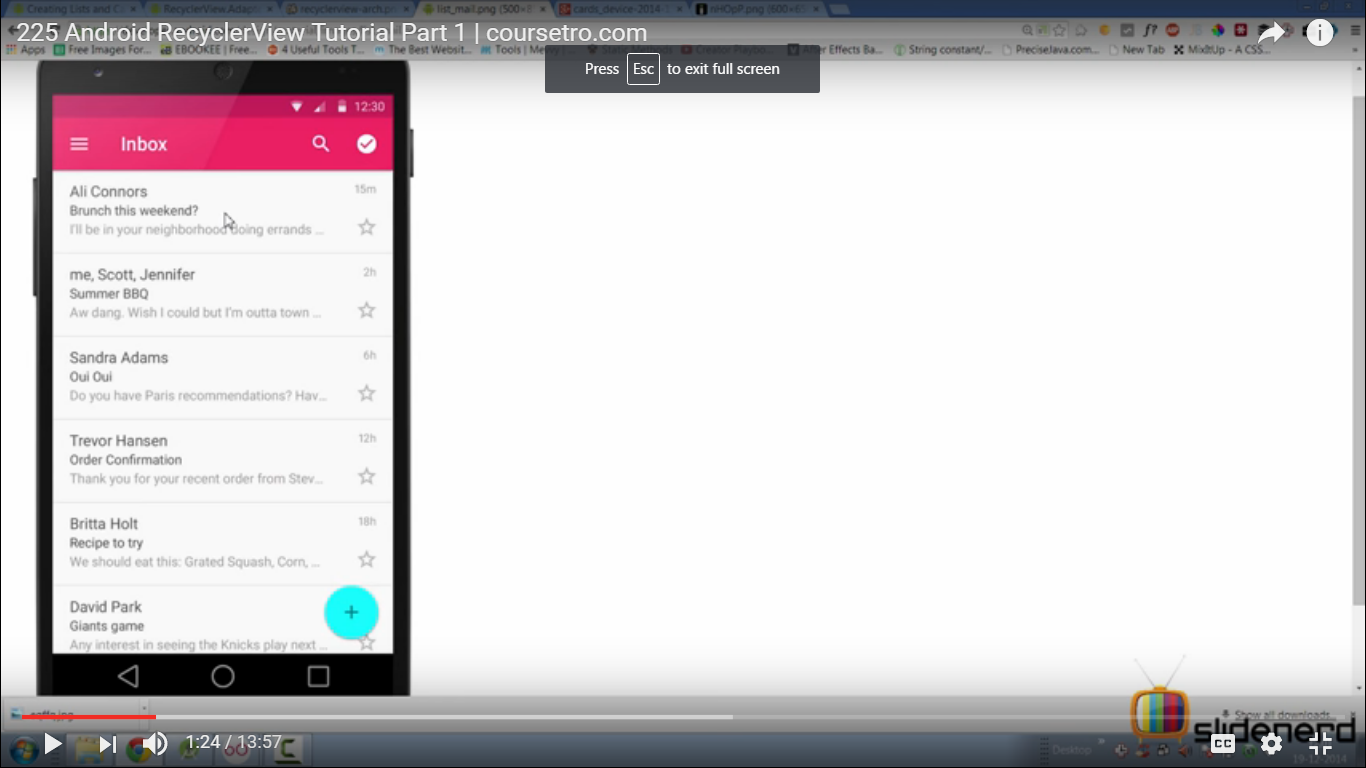
[RecyclerView](https://developer.android.com/reference/android/support/v7/widget/RecyclerView.html) provides these built-in layout managers:

* [LinearLayoutManager](https://developer.android.com/reference/android/support/v7/widget/LinearLayoutManager.html) shows items in a vertical or horizontal scrolling list.
* [GridLayoutManager](https://developer.android.com/reference/android/support/v7/widget/GridLayoutManager.html) shows items in a grid.
* [StaggeredGridLayoutManager](https://developer.android.com/reference/android/support/v7/widget/StaggeredGridLayoutManager.html) shows items in a staggered grid.

c)



* This diagram:
* Data model is one single item inside recycler view. -> img+textview.



* Adapter is going to pull all the single items and going to make appearance using view holder.
* View holder is xml layout for this single item (converted/inflated into java code). Going to maintain if data item is going to be inflated every time.
* Recycler view is going to use VH for displaying every item present.
* Layout Manager : tells how each item would be placed. Here one below the other.
* Item Animater : lets you do animation on recycler view.(eg add item , remove item , swipe item , disappear item , collapse recycle view)
* Item decorater : sections separate.

Code

Xml:

* <android.support.v7.widget RecyclerView …>

Java code:

* private RecyclerView recyclerview;
* onCreateView

{

View layout=inflater.inflate(R.layout.layout1,container,false)

recyclerView=(RecyclerView)layout.findViewById(R.id.layout);

return layout;

}

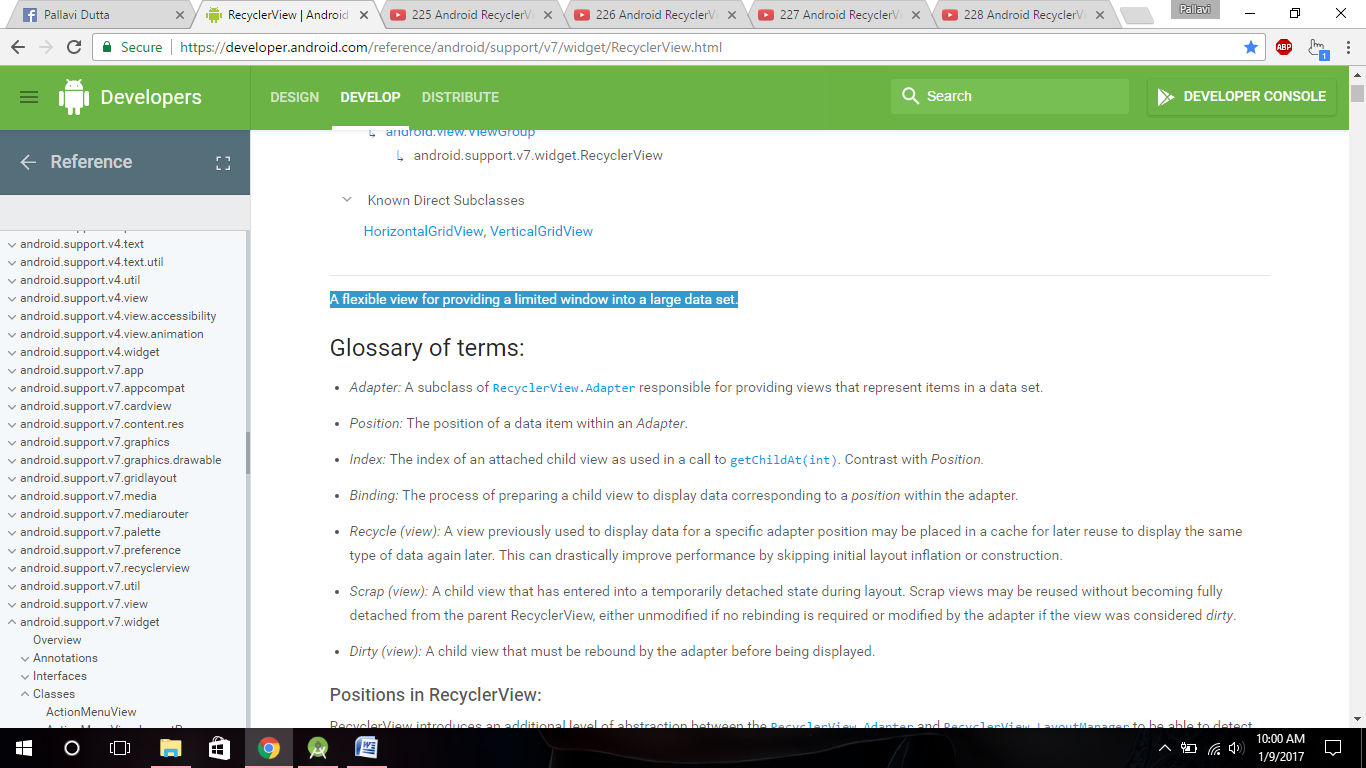
* Link

<https://developer.android.com/reference/android/support/v7/widget/RecyclerView.html>

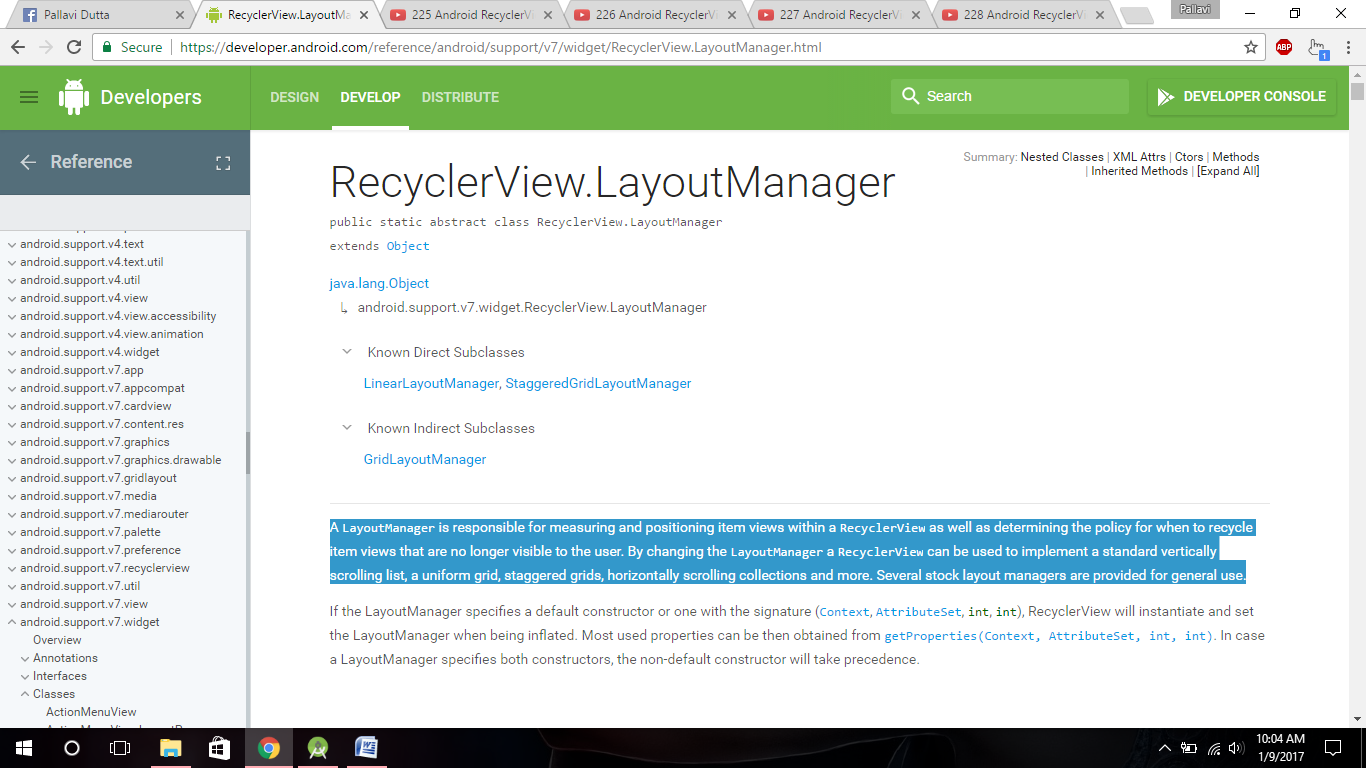


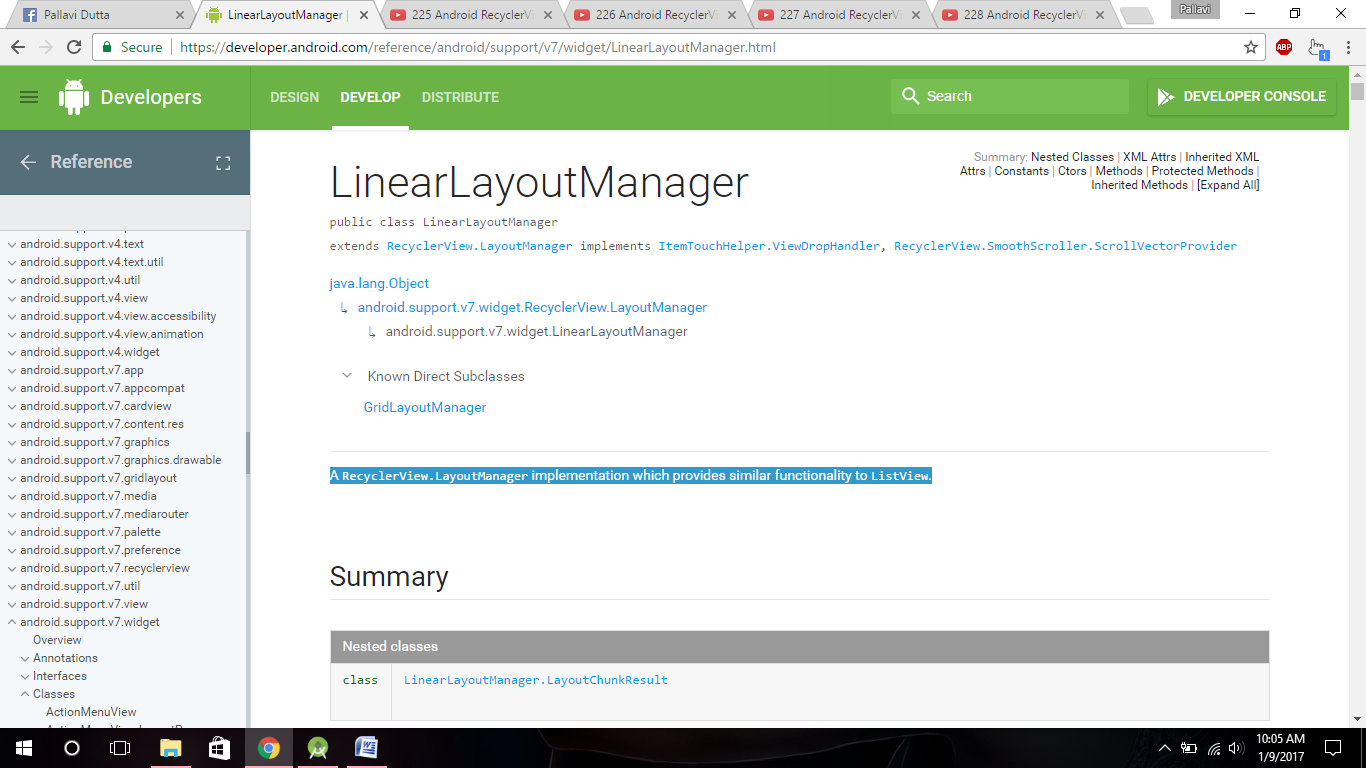
1

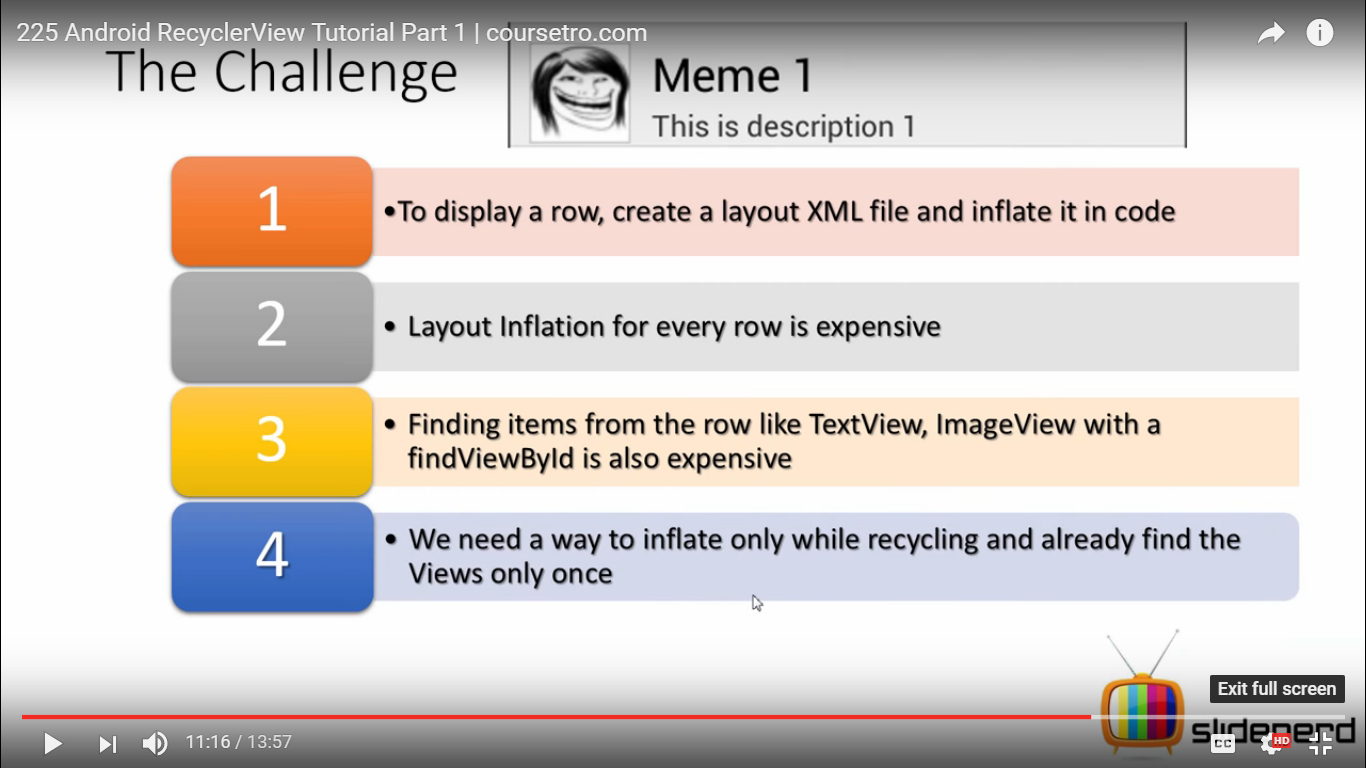
* extends from ViewGroup ->layout(not list)



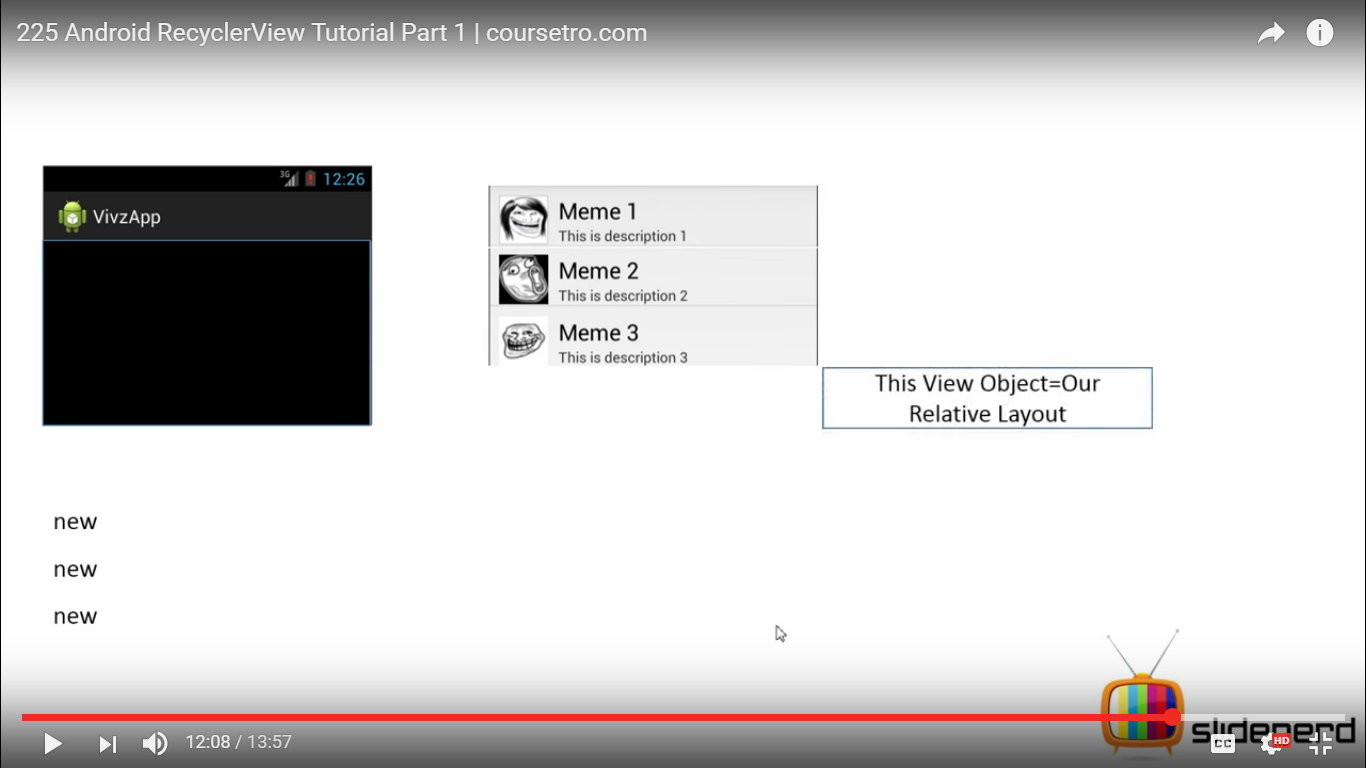
* Link -> <https://developer.android.com/reference/android/support/v7/widget/RecyclerView.LayoutManager.html>



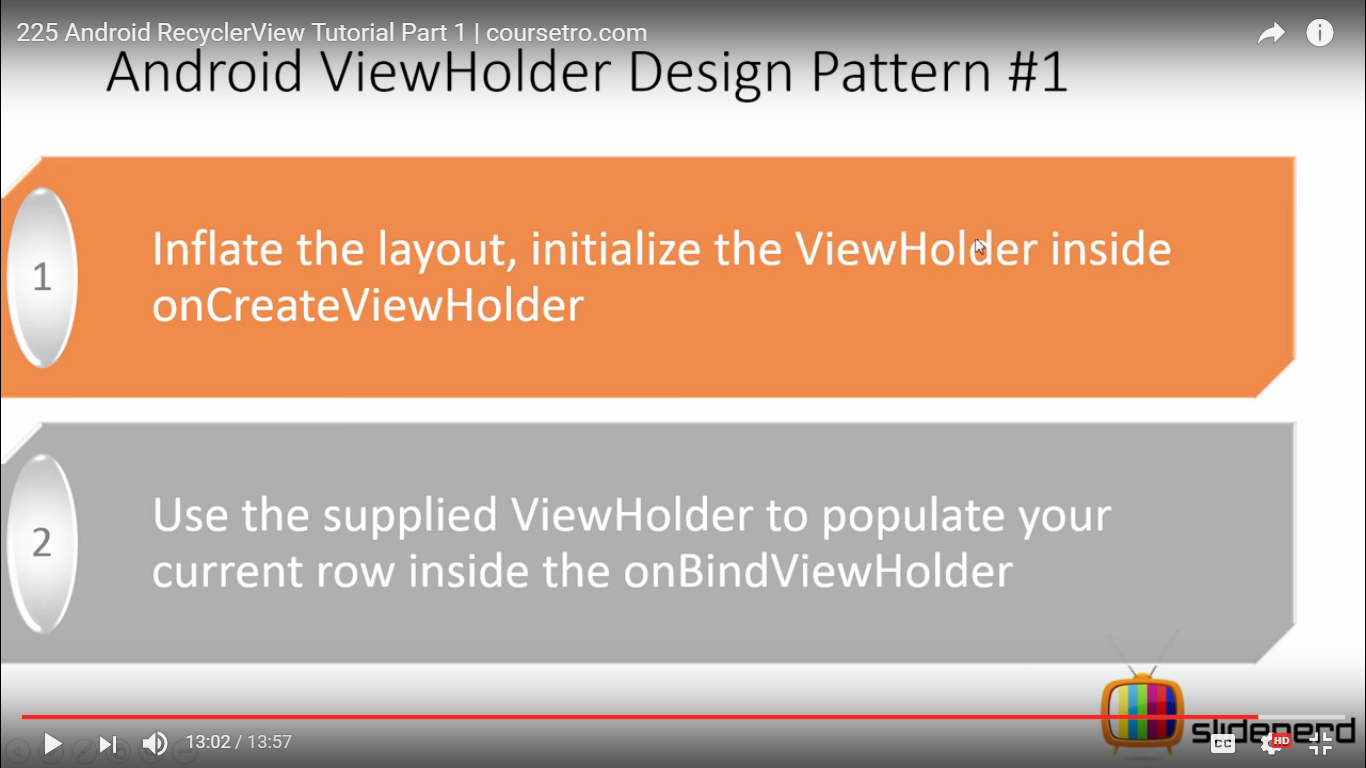




* To dislay a row(img+textview) , create layout inflation every time and findViewById every time.



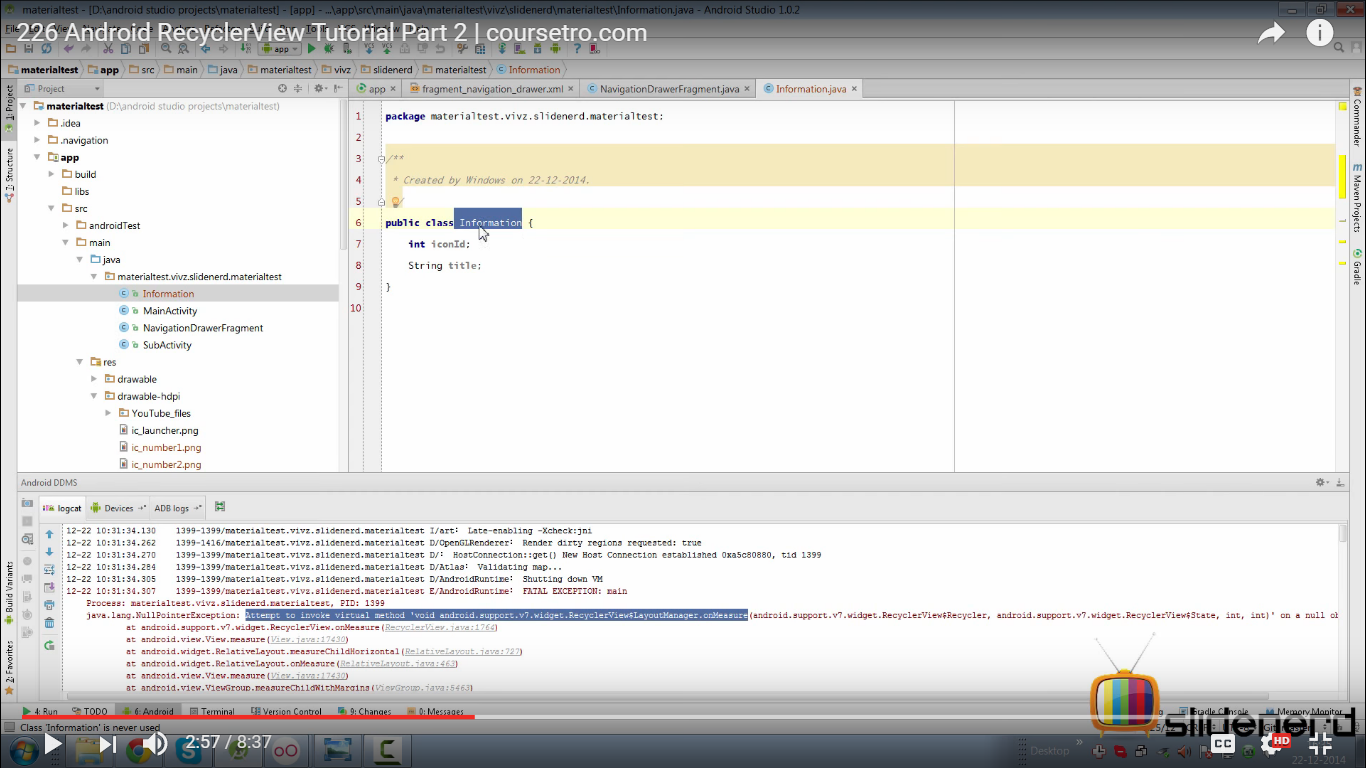
* once there are 3 rows each time new object created .next 4th row is generated with the help of RecyclerView.When you scroll recyclerView is going to generate a previously cached viewHolder
* Don’t create VH object every time. Don’t call findViewById every time.Do it for no of times row fits in screen then recycle previous rows.



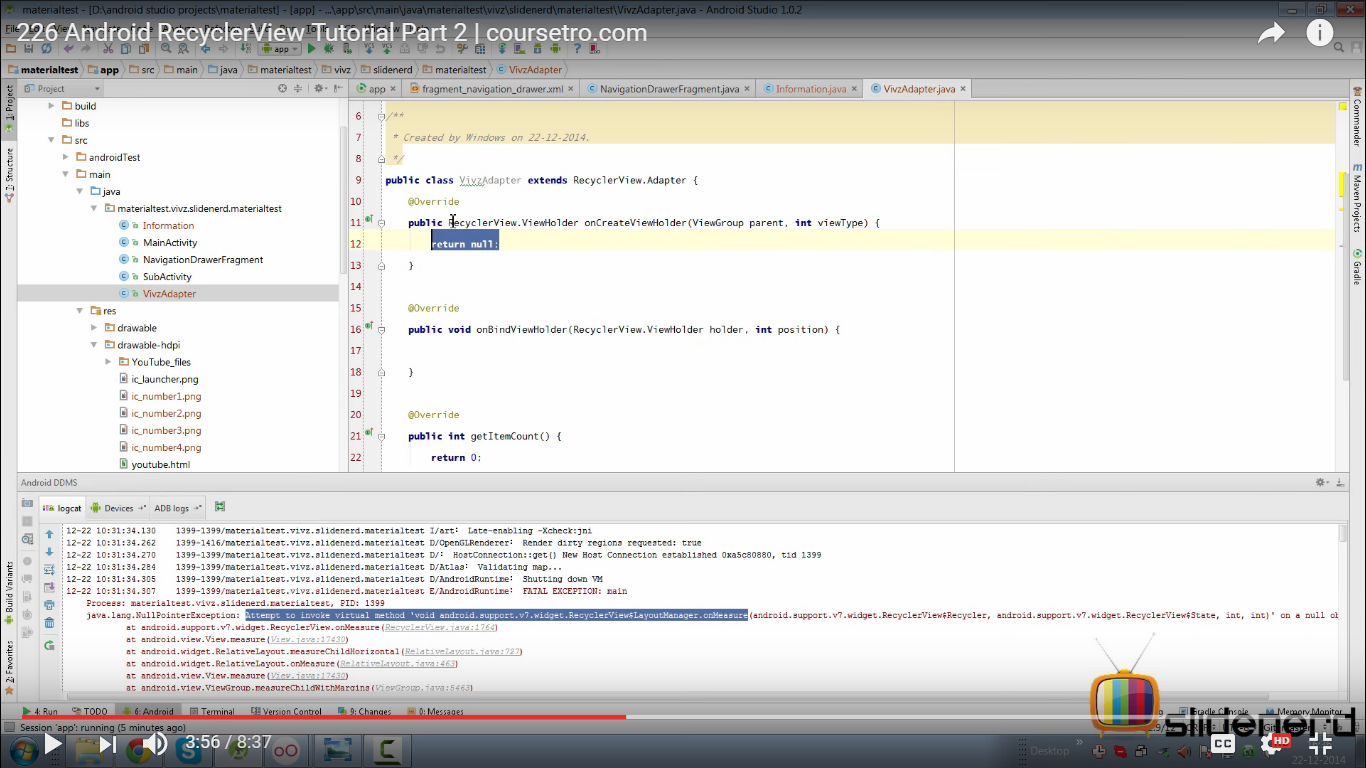
NEXT VIDEO

<https://www.youtube.com/watch?v=A5cUZlUnRUw>

* Next video : Adapter,onCreateViewHolder,onBindViewHolder.
* if recyclerView is used then it crashes if there is no LayoutManager.
* Create a java Class ->Info ->variables img , text .



* Java class :- Adapter



* …Adapter extends RecycleView Adapter -> 3 methods implement (onCreateView,onBindViewHolder,getItemCount() )
* \*\*recyclerView.Adapter takes a generic argument ->VH
* recyclerView.ViewHolder(present in 2nd method,required in first method)

<https://developer.android.com/reference/android/support/v7/widget/RecyclerView.ViewHolder.html>

A ViewHolder describes an item view and metadata about its place within the RecyclerView.

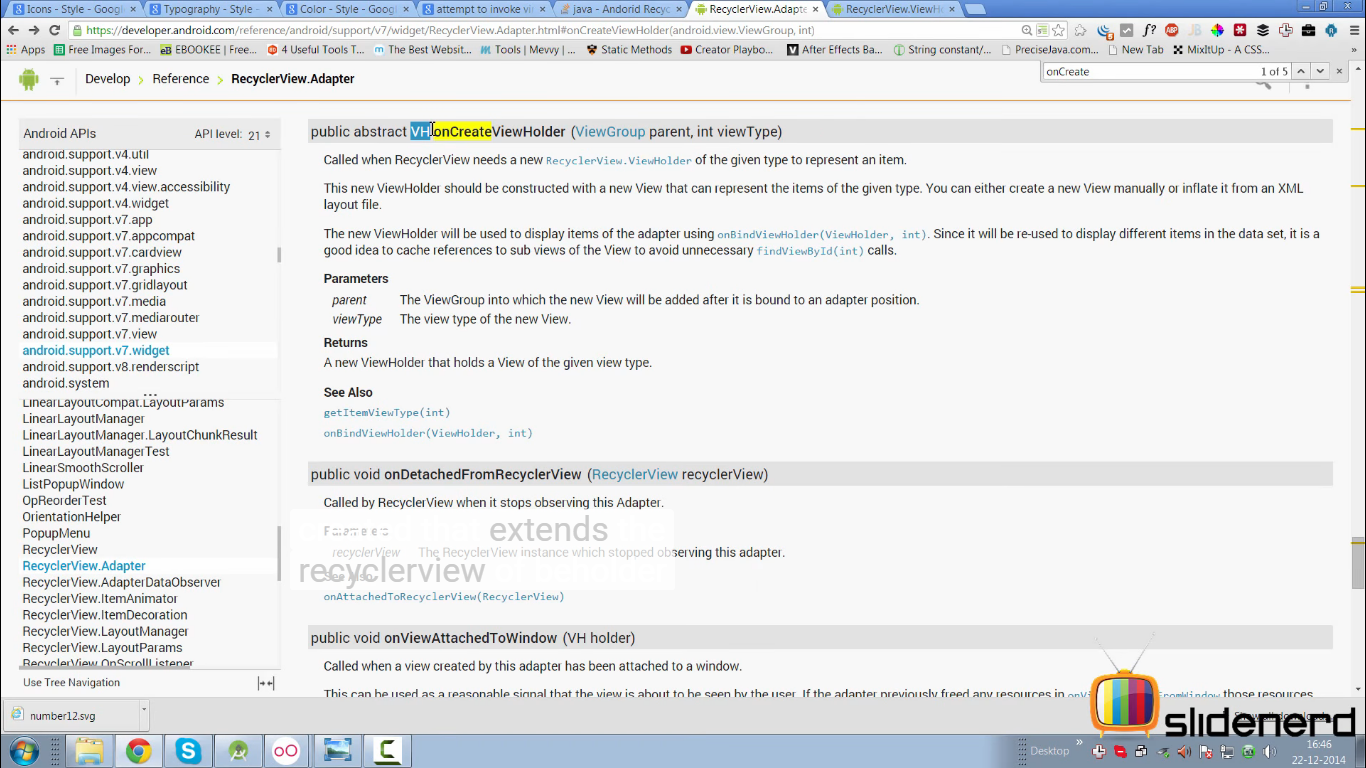
* class MyViewHolder extends RecyclerView.ViewHolder{

constructor MyViewHolder(view textView)

}

* \*\*Now replace RecyclerView.ViewHolder with MyViewHolder

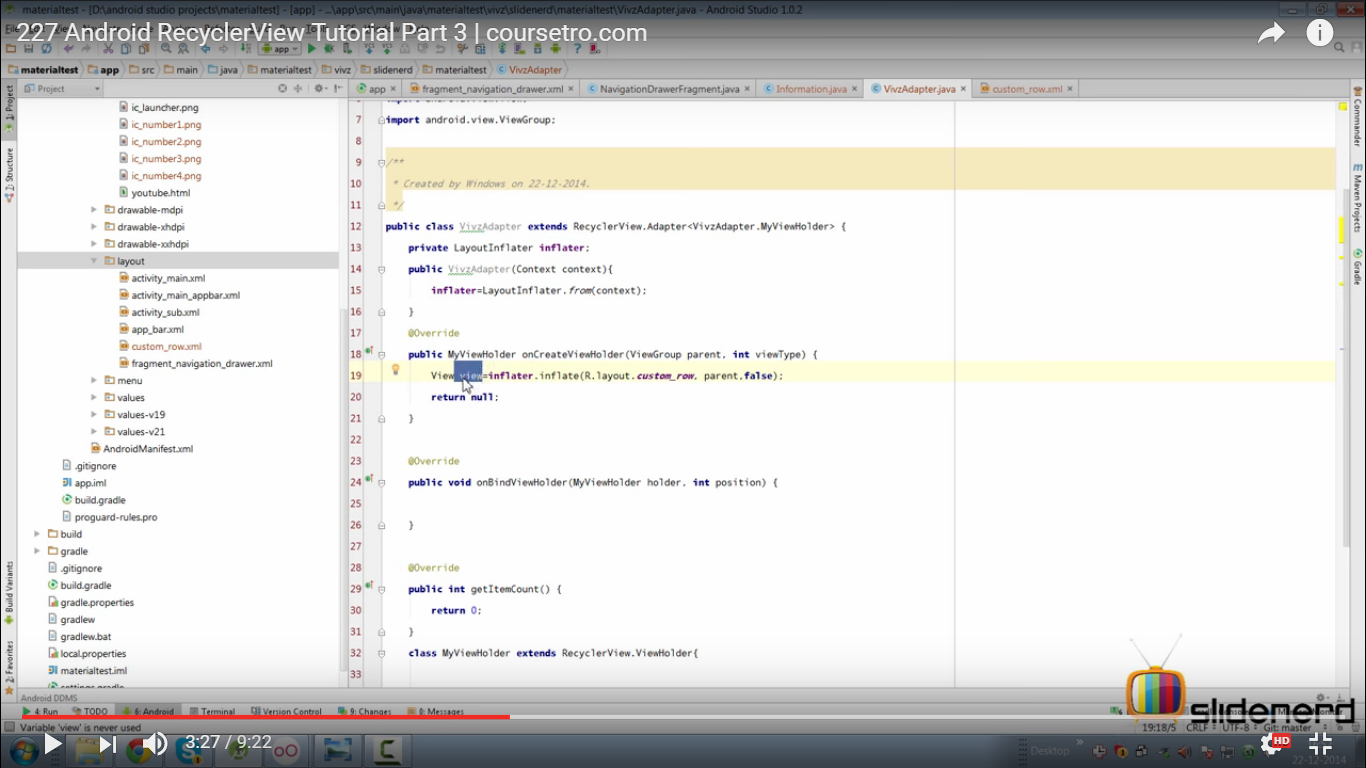
onCreateViewHolder link :->



Idea behind RecycleView is find things(findViewById) once and use the cache again and again.

The new ViewHolder should be connected with a new View.

New view can be created manually or inflate it from an XML layout file.



* private LayoutInflater inflater;

public VAdapter(Context context){

inflater.LayoutInflater.from(context);

}

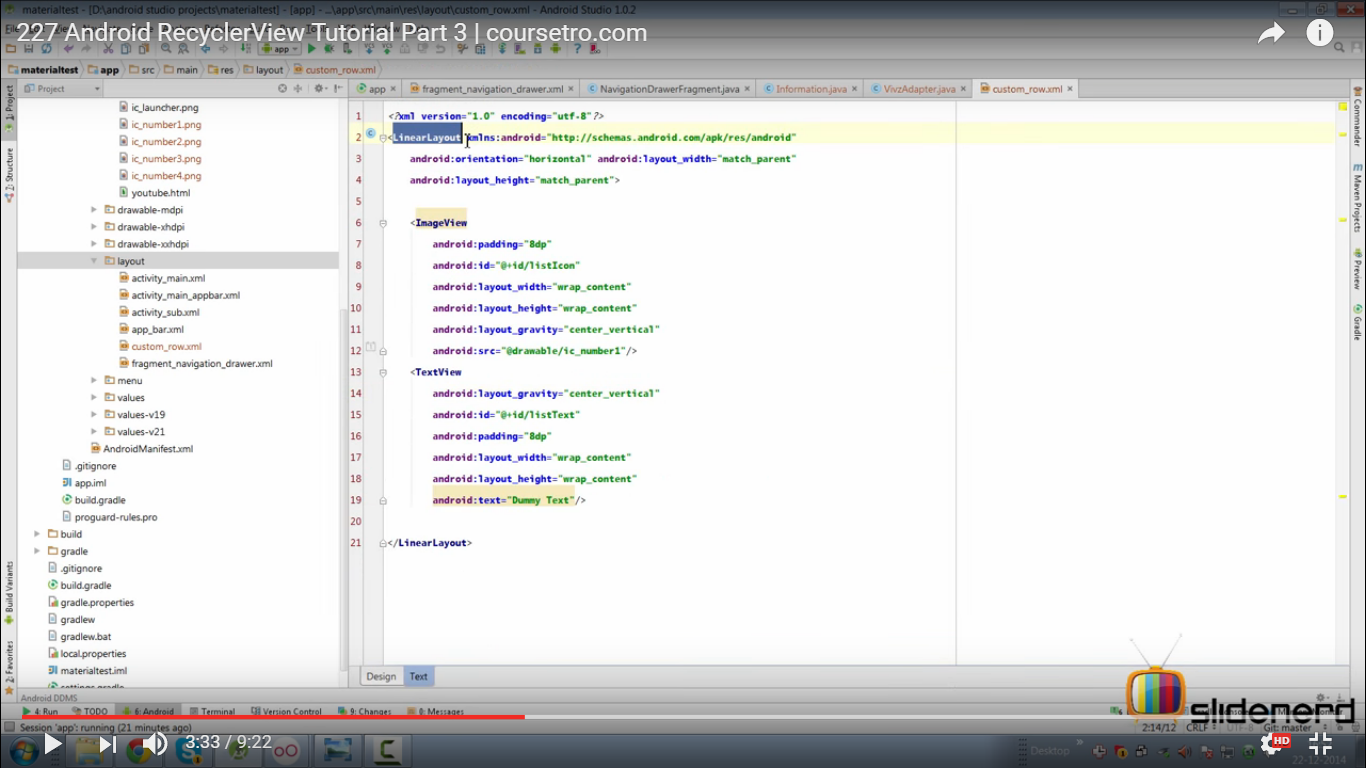
* onCreateViewHolder()

finds view

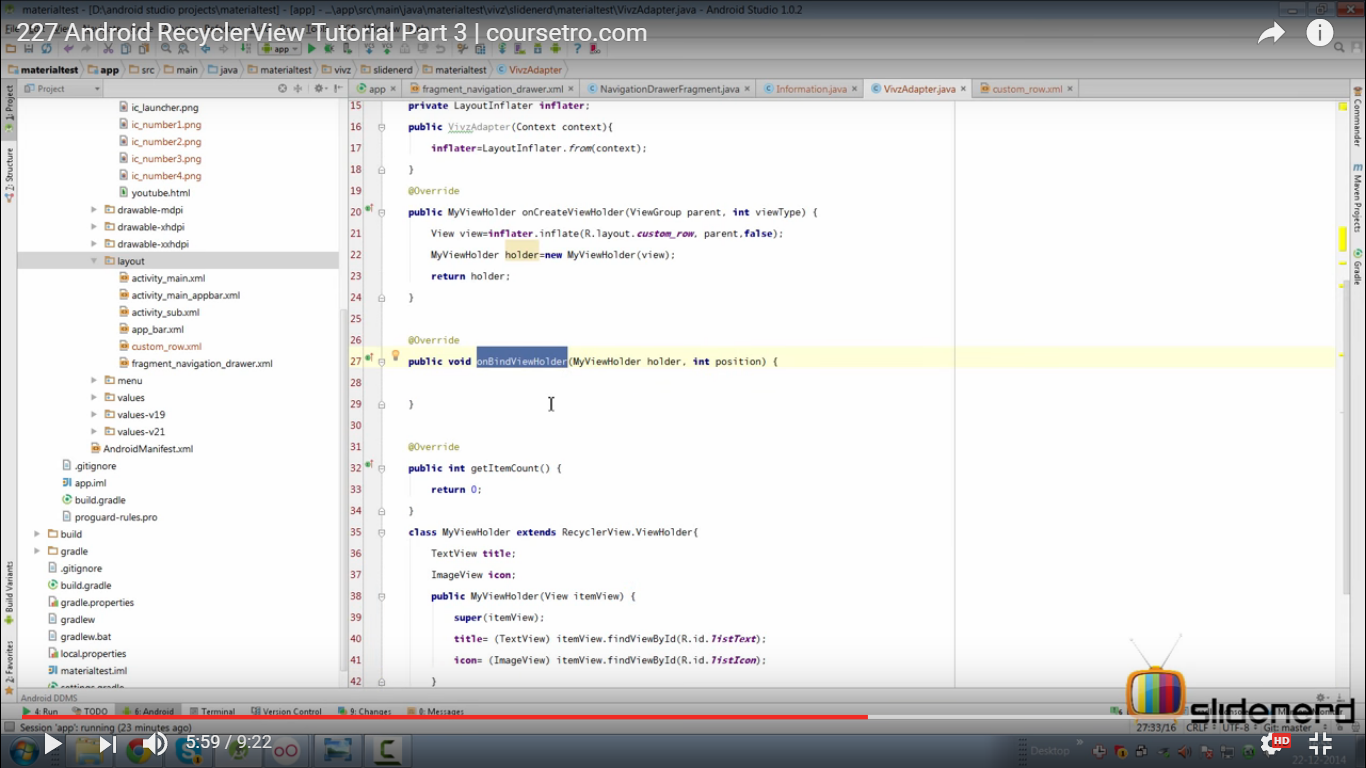
* onBindViewHolder()

sets view

* getItemCount()



view here is parent root -> LinearLayout where we have ImageView and TextView.



* Code:

**package** com.kpf.sujeet.chat.Adapter;  
  
**import** android.content.Context;  
**import** android.support.v7.widget.RecyclerView;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.TextView;  
  
**import** com.android.volley.toolbox.ImageLoader;  
**import** com.android.volley.toolbox.NetworkImageView;  
**import** com.kpf.sujeet.chat.Models.User;  
**import** com.kpf.sujeet.chat.R;  
**import** com.kpf.sujeet.chat.Utils.AppController;  
  
**import** java.util.List;  
  
*/\*\*  
 \* Created by SUJEET on 1/7/2017.  
 \*/***public class** ChatListRecyclerAdapter **extends 1**.RecyclerView.Adapter<ChatListRecyclerAdapter.**3**MyViewHolder> {  
  
 ImageLoader **imageLoader**;  
 **5** Context **context**;  
 List<User> **userList**;  
 **public** ChatListRecyclerAdapter(Context contex, List<User> userList){  
 **this**.**context** = contex;  
 **this**.**userList**=userList;  
 **imageLoader** = AppController.*getInstance*().getImageLoader();  
 }  
  
  
 **2.** **public class** MyViewHolder **extends** RecyclerView.ViewHolder`{  
  
 NetworkImageView **network\_chat\_image**;  
 TextView **txt\_chat\_name**;  
  
 **public** MyViewHolder(View itemView) {  
 **super**(itemView);  
 **network\_chat\_image** = (NetworkImageView)itemView.findViewById(R.id.***network\_chat\_image***);  
 **txt\_chat\_name** = (TextView)itemView.findViewById(R.id.***txt\_chat\_user\_name***);  
 }  
 }  
 @Override  
 **public** MyViewHolder onCreateViewHolder(ViewGroup parent, **int** viewType) {  
 **4.** View view = LayoutInflater.*from*(parent.getContext()).inflate(R.layout.***chat\_list\_layout***, parent, **false**);  
 **return new** MyViewHolder(view);  
 }  
  
 @Override  
 **6** **public void** onBindViewHolder(MyViewHolder holder, **int** position) {  
  
 User user = **userList**.get(position);  
  
 **if**(**userList**.get(position).**photoUrl**.equals(**""**)) {  
 holder.**network\_chat\_image**.setDefaultImageResId(R.drawable.***defaultimg***);  
 }**else**{  
 holder.**network\_chat\_image**.setImageUrl(**userList**.get(position).**photoUrl**, **imageLoader**);  
 }  
 holder.**txt\_chat\_name**.setText(user.**name**);  
 }  
  
 @Override  
 **7** **public int** getItemCount() {  
 **return userList**.size();  
 }  
  
}

8ChatFragment

**package** com.kpf.sujeet.chat.Fragments;  
  
  
**import** android.os.Bundle;  
**import** android.support.v4.app.Fragment;  
**import** android.support.v7.widget.DefaultItemAnimator;  
**import** android.support.v7.widget.LinearLayoutManager;  
**import** android.support.v7.widget.RecyclerView;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.TextView;  
  
**import** com.google.firebase.auth.FirebaseAuth;  
**import** com.google.firebase.database.DataSnapshot;  
**import** com.google.firebase.database.DatabaseError;  
**import** com.google.firebase.database.FirebaseDatabase;  
**import** com.google.firebase.database.Query;  
**import** com.google.firebase.database.ValueEventListener;  
**import** com.kpf.sujeet.chat.Adapter.ChatListRecyclerAdapter;  
**import** com.kpf.sujeet.chat.Models.User;  
**import** com.kpf.sujeet.chat.R;  
  
**import** java.util.ArrayList;  
**import** java.util.Iterator;  
**import** java.util.List;  
  
  
  
*/\*\*  
 \* A simple {****@link*** *Fragment} subclass.  
 \*/***public class** ChatListFragment **extends** Fragment {  
 RecyclerView **recyclerview**;  
 List<User> **userList**;  
  
  
 **public** ChatListFragment() {  
 *// Required empty public constructor* }  
  
  
 @Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
  
 **userList** = **new** ArrayList<User>();  
  
 **recyclerview** = (RecyclerView) inflater.inflate(R.layout.***fragment\_chatlist***, container, **false**);  
  
 RecyclerView.LayoutManager layoutManager = **new** LinearLayoutManager(getContext());  
 **recyclerview**.setLayoutManager(layoutManager);  
 **recyclerview**.setItemAnimator(**new** DefaultItemAnimator());  
  
  
 **return recyclerview**;  
 }  
  
 @Override  
 **public void** onStart() {  
 **super**.onStart();  
 **userList**.clear();  
 **9** Query query= FirebaseDatabase.*getInstance*().getReference();  
 query.addValueEventListener(**new** ValueEventListener() {  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot) {  
 Iterator iterator=dataSnapshot.getChildren().iterator();  
  
 **while** (iterator.hasNext()){  
  
 DataSnapshot dataSnapshot2=(DataSnapshot)iterator.next();  
 **if** (dataSnapshot2.getKey().equals(**"users"**)){  
 Iterator iterator1 = dataSnapshot2.getChildren().iterator();  
 **while** (iterator1.hasNext()){  
 DataSnapshot dataSnapshot3 = (DataSnapshot) iterator1.next();  
 Iterator iterator2 = dataSnapshot3.getChildren().iterator();  
 User user=**new** User();  
 user.**uid** = dataSnapshot3.getKey();  
 **while** (iterator2.hasNext()){  
 DataSnapshot snapshot = (DataSnapshot) iterator2.next();  
 **if** (snapshot.getKey().equals(**"name"**)){  
 user.**name** = snapshot.getValue().toString();  
 }  
 **if** (snapshot.getKey().equals(**"photoUrl"**)){  
 user.**photoUrl** = snapshot.getValue().toString();  
 }  
 **if**(snapshot.getKey().equals(**"email"**)){  
 user.**email** = snapshot.getValue().toString();  
 }  
 **if**(snapshot.getKey().equals(**"contact\_no"**)){  
 user.**mobno** = snapshot.getValue().toString();  
 }  
 **if**(snapshot.getKey().equals(**"country"**)){  
 user.**country** = snapshot.getValue().toString();  
 }  
 }  
 **userList**.add(user);  
 }  
 }  
 }  
 **10** ChatListRecyclerAdapter chatListRecyclerAdapter = **new** ChatListRecyclerAdapter(getActivity(), **userList**);  
 **recyclerview**.setAdapter(chatListRecyclerAdapter);  
 }  
  
 @Override  
 **public void** onCancelled(DatabaseError databaseError) {  
  
 }  
 });  
 }  
}