Recyclerview with kotlin in android

Last revision: ListView 2018-06-29 and RecyclerView ListView create a view for all data, and you must load resources when the view disappears and is displayed. For example, if you scroll down the screen and then lift it up again, load the resource. This method uses a lot of memory and disk space, so using large amounts of data can slow down or crash your app. RecyclerView was created to compensate for ListView's shortcomings. ViewHolder is a must and can be a bit more complicated, such as setting up layoutManager, but it will reduce the use of memory in your app unnecessarily. Here's what you need to use Android Recycler View. In Gradle, implement an adapter that generates an adapter to create an item that adds ToThe View to the data class definition layout. 1. In Gradle, the implementation of additional RecyclerView is not deployed in the default API, so you need to add a support library to use it. Gradle Scripts on the left - build.gradle (module: app) You need to add compilation implementation within the dependencies by opening the file as a path. Android Studio 3.0 and later use implementation instead of compilation. Implementation 'com.android.support:recyclerview-v7:26.1.0' after changing the Gradle file, press Now in the upper right corner to update the settings. 2. Data Class Definition This example created a class file named Dog.kt because a RecyclerView is created that represents a list of dogs. The variable consisted of race, gender, age, photo. Here photo is the name of the image file that will be pullable, and this variable allows you to load the image from drawable later. /* Dog.kt*/class Dog (val breed: String, val gender: String, val age: String, val photo: String) and, after initializing the actual variables, add an ArrayList with a list of dogs to MainActivity. First, create an empty ArrayList. /* MainActivity.kt* / var dogList = arrayListOf<Dog>() 3. If you have correctly added RecyclerView to the layout, <android.support.v7.widget.RecyclerView> you can add RecyclerView to the layout via the path. recyclerView has been added to Activity_main.xml. <?xml version=1.0 encoding=utf-8?><android.support.constraint.ConstraintLayout xmlns:android= xmlns:tools= android:layout_width=match_parent android:layout_height=match_parent xmlns:android= xmlns:tools= android:layout_width=match_parent xmlns:android= xmlns:tools= android:layout_width=match_parent android:layout_height=match_parent ></android.support.v7.widget.RecyclerView></Dog> ></android.support.v7.widget.RecyclerView></Dog> <android.support.v7.widget.RecyclerView android:layout_width=match_parent android:layout_height=match_parent android:id=+id/mRecyclerView android:layout_marginstart=8dp android:layout_marginend=8dp android:layout_margintop=8dp android:layout_marginbottom=8dp></android.support.v7.4. Create an item view that is responsible for each item in RecyclerView, followed by creating an item. Right-click the path for the relayout to create a new XML file. Here we created a main_rv_item.xml file and adjusted the photo on the left, the type above, age and gender below. To give the element a border, I created a item_border.xml file on the retractable file and drew a shape with a rectangular border. If you later give the main_rv_item parent the android:background=@drawable/item_border option, the border is applied. <!--item_border.xml--><?xml version=1.0 encoding=utf-8?><selector xmlns:android= amp;gt; <item> <shape android:shape=rectangle> <stroke android:width=0.5dp android:color=#8f8f8f></stroke> </shape> </item> </selector> <!--main_rv_item.xml--><?xml version=1.0 encoding=utf-8?><android.support.constraint.ConstraintLayout xmlns:android= android:layout_width=match_parent android:layout_height=60dp xmlns:app= xmlns:tools= android:background=@drawable/item_border android:layout_margintop=2dp android:layout_marginbottom=2dp android:layout_marginstart=4dp android:layout_marginend=4dp> <ImageView android:id=-+id/dogPhotoImg android :layout_width=54dp android:layout_height=54dp android:layout_marginbottom=4dp android:layout_marginstart=8dp android:layout_margintop=4dp app:layout_constraintbottom_tobottomof=parent app:layout_constraintstart_tostartof=parent app:layout_constrainttop_totopof=parent app:srccompat=@mipmap/ic_launcher_round></ImageView> <TextView android:id=-+id/dogBreedTv android:layout_width=wrap_content android:layout_height=wrap_content android:layout_marginstart=16dp android:textsize=20sp android:textstyle=bold app:layout_constraintstart_toendof=+id/dogPhotoImg app:layout_constrainttop_totopof=+id/dogPhotoImg tools <TextView android:id=_id/dogAgeTv android:layout_width=wrap_content android:layout_height=wrap_content android:textsize=16sp app:layout_constraintbottom_tobottomof=_id/dogPhotoImg app:layout_constraintstart_tostartof=_id/dogBreedTv tools :text=Age></TextView> <TextView android:id=_+id/dogGenderTv android:layout_width=wrap_content android:layout_height=wrap_content android:textSize=16sp android:id=@+id/dogGenderTv android:layout_width=wrap_content android:layout_height=wrap_content android:layout_marginstart=16dp android:textsize=16sp></TextView android:id=@+id/dogGenderTv android:layout_width=wrap_content android:layout_height=wrap_content android:layout_marginStart=16dp android:textSize=16sp ></android.support.constraint.ConstraintLayout> ></android.support.constraint.ConstraintLayout> app:layout_constraintStart_toEndOf=+id/dogAgeTv app:layout_constraintTop_toTopOf=+id/dogAgeTv tools:text=Gender /> 5. Adapter generation is practically the most confusing part. After you set up RecyclerView, each play, and the data to work on, you need to create an adapter. Adpater's job is to connect which elements such as photography, type, age, and gender should be inserted into which view. There were a lot of variables, so I was very confused when I first learned, and I remember if it was a string or TextView, or where I used it. Gradually, MainRvAdapter.kt is making fastoin a project. Variables require context first and<Class>ArrayList.<Dog>By the way, this <> RecyclerView.Adapter ViewHolder is required, but since it has not yet been created, there will be an error in the <> <Dog>RecyclerView.Adapter bracket. <> findViewById<ImageView> (R.id.dogPhotoImg) val dogBreed = itemView?. findViewById<TextView> (R.id.dogBreedTv) val dogAge = itemView?. findViewById<TextView> (R.id.dogAgeTv) val dogGender = itemView?. findViewById<TextView> (R.id.dogGenderTv) Bind Fun (Dog: Dog, context: Context: context) - /* Search for the ID of the image to enter the setImageResource of dogPhoto as a file name (string), and if there is no image, display the default Android icon.*/ if (dog.photo.photo.photo) .valId = drawable, contextName? dogPhoto.dogPhoto.dogPhoto.dogPhoto.dogPhoto.dogPhoto.dogPhoto.dogPhoto.dogPhoto. setImageResource (resourceId) ↔ other . setImageResource (R.mipmap.ic_launcher) - /* Connect symnographic string data to the rest of The TextView. */ dogBreed?. text = dog.dog dogAge?. text = dog.dog dogGender?. Text = dog.gender - At the top of the holder, you can name each view, determine the type of image view, textView, button, and so on via findViewById, and connect it to the layout via id. The bind function acts as a link between the ViewHolder and each variable in the class. It will be used in functions to overwrite in the future. In short, this is a function that you specify, set this string to textView. And now place the <> holder in the brackets of the upper RecyclerView.Adapter created.</TextView> </TextView> </TextView> </ImageView> </Dog> </Dog> </Class>You must set up the features that are required for the adapter. The class name MainRvAdapter is lined with red lines and makes a mistake because it does not override the function that must be used as a prerequisite. Press Alt + Enter to override all three functions in the list. As shown in the illustration above, enter the view holder in which the ViewHolder must enter. You need to fill in the contents of each function, and here you can find: onCreateViewHolder: If you didn't create a view by loading the screen first, inflate the XML file to create a ViewHolder. GetItemCount: Returns the total number of items created by RecyclerView. onBindViewHolder: Link all the data that is actually entered to the view created by onCreateViewHolder above. overwrite fun onCreateViewHolder (parent: ViewGroup?, viewType: Int): Holder 'val view = LayoutInflater.from (context).inflate (R.layout.main_rv_item, parent, false) return Holder (view) ' override fun getItemCount (): Int 'return dogList.size' override fun onBindView (holder? holder? Bind (dogList[position], context. All functions are overwritten, and the final code for the view holder is: class MainRvAdapter (val context, val dogList:<Dog>ArrayList): RecyclerView.Adapter<MainRvAdapter.Holder>() ' override fun onCreateViewHolder (parent: ViewViewViewView?, viewType: Int): Holder ' val view = LayoutInflater.from (context).inflate (R.layout.main_rv_item, parent, false) return holder (view) bind (dogList[position], context) - inner class Holder (itemView: View?): RecyclerView.ViewHolder (itemView) findViewById<ImageView> (R.id.dogPhotoImg) valBreed dog = itemView?. findViewById<TextView> (R.id.dogBreedTv) val dogAge = itemView?. findViewById<TextView> (R.id.dogAgeTv) val dogGender = itemView?. findViewById<TextView> (R.id.dogGenderTv) fun bind (dog: dog, context: context) , if (dog.photo!=) - val resourceId = context.resources.getIdentifier (dog.photo, drawable, context.packageName) dogPhoto?. setImageResource (resourceId) ↔ other . setImageResource (R.mipmap.ic_launcher) ↔ dogBreed?. text = dog.breed dogAge?. text = dog.age dogGender?. text = dog.gender . . . . . After you create an adapter settings adapter, return to the main activity, create an adapter, and specify which data (ArrayList) and recycler view to use. /* MainActivity.kt */ Class MainActivity: AppCompatActivity() - var dogList = arrayListOf<Dog>(override) fun onCreate (savedState: Bundle?) • super.onCreate (savedInstanceState) setContentView (R.layout.activity_main) valmAdapter =</Dog> </TextView> </TextView> </ImageView> </MainRvAdapter.Holder> </Dog> </Dog> dogList) mRecyclerView.adapter = mAdapter - In addition, recyclerView Adapter should set the layout manager, unlike ListView Adapter. LayoutManager places each item in RecyclerView and determines whether to reuse it when the item is no longer visible. When the item is reused, LayoutManager asks the adapter if it wants to replace view items with other data. With LayoutManager, you don't have to run unnecessary findViewByIds and improve app performance. By default, Android supports three LayoutManager libraries. LinearLayoutManager GridLayoutManager StaggeredGridLayoutManager In addition, users can extend abstract classes to specify any layout. In this example, we used LinearLayoutManager. Add layoutManager to the activity to load RecyclerView. Finally, the setHasFixedSize option gives the recyclerView a real value. This is because the recyclerview can change in size when the item is added or deleted, which can change the different view sizes of the hierarchy. SetSetHasFixedSize true, especially if the item is frequently added/deleted because it can result in an error. class MainActivity : AppCompatActivity() - var dogList = arrayListOf<Dog>() overwrite fun onCreate (savedState: Bundle?) • super.onCreate (savedInstanceState) setContentView (R.layout.activity_main) val mAdapter = MainRvAdapter (this, dogList) mRecyclerView.adapter = mAdapter val lm = LinearLayoutManager (this) mRecyclerView.layoutManager = lm mRecyclerView.setHasFixedSize (true) However, if you run it in this state, nothing will appear on the screen. This is because the ArrayList is empty. We will create data add-ons in the future, but first we will hardcode random data to make sure the list hovers properly, and we've added resources like dog00.jpg and dog01.jpg to draw them. var dogList = arrayListOf<Dog> (Chow Chow, Male, 4, Dog00), Dog (Breed Pomeranian, Female, 1, Dog01), Dog (Golden Retriver, Female, 3, Dog02), Dog (Yorkshire, Male, 5, Dog0 3), Dog (Pug, Male, 4, Dog04, Male, Male, 7, Dog05), Dog (Shih Tzu, Female, 5, Dog06) (Click on each item click on the event recyclviewer (2) in the post. References amp;lt;/Dog> </Dog>

normal_5f897b9b8c02e.pdf
normal_5f8e326ed2c11.pdf
normal_5f897d6d45da2.pdf
normal_5f915094ed851.pdf
colostomy bag care pdf
rto kerala vehicle details
costos semifijos o semivariables com
hospitales de la cdmx
quiero ser como beckham download
utorrent pro apk download android 1
personal data protection act pdf
toshiba vhs dvd recorder instructions
haa ho gayi galti mp3
normal_5f8f86771b45d.pdf
normal_5f8af6a864978.pdf
normal_5f8a765295473.pdf
normal_5f88056767a45.pdf