**CardView简介**

* CardView继承自FrameLayout类。
* CardView是一种卡片视图，主要是以卡片形式显示内容。

**CardView功能**

* CardView实现在一个卡片布局中显示相同的内容，卡片布局可以设置圆角和阴影，还可以布局其他的View。
* CardView即可作为一般的布局使用，也可以作为ListView和RecyclerView的Item使用。

**CardView何时使用**

* 需要显示层次性的内容，可以考虑使用。
* 需要显示列表或网格时，可以考虑使用。

**CardView位置**

* 包名：android.support.v7.widget.CardView
* 文件地址有两个
  + android-sdk/extras/android/m2repository/com/android/support/cardview-v7
  + android-sdk/extras/android/support/v7/cardView

**CardView引用**

**Android Studio**

dependencies {

compile 'com.android.support:cardview-v7:23.1.1'

}

**Eclipse**

* 在android-sdk/extras/android/support/v7/cardview目录下面有libs，里面有jar包，引用此jar包。
* 在android-sdk/extras/android/m2repository/com/android/support/cardview-v7目录下根据版本号23.1.1目录可以找到一个名为cardview-v7-23.1.1.aar的文件。解压此文件里面有classes.jar，引用此jar包。
* 针对找不到目录，可以打开Android SDK Manager把最新的资源更新下来即可。

**在此推荐使用Android Studio开发Android项目**

**开发中值得注意的细节**

* 对低版本SDK的兼容（低于Android L版本）
  + 针对Android L以下的版本，对CardView添加了一个Elevation的元素，即XML中的app:cardElevation和代码中的setCardElevation。
  + 对于在低版本中，设置了CardElevation后，CardView会自动留出空间供阴影显示，但对于Android L版本，就需要手动设置Margin边距来预留空间，这样的结果就是在Android 5.0以上的手机上可以正常显示，但对于Android 4.4.x的手机上就发现边距很大，导致浪费了屏幕空间。
  + 解决上面问题，需要我们做适配。可以在/res/value和/res/value-v21分别创建dimens.xml文件，在dimens.xml里，随意命名，对于Android 5.0以上的设置数值0dp，对于Android 5.0以下的设置数值（根据实际需求）。这样就解决低版本中边距过大或者视觉效果不好的问题了。
* 对低版本SDK的兼容（低于Android L版本）setElevation的问题
  + 由于缺少一些系统API（如 RenderThread），CardView中的Elevation兼容实现并不完美，和真正的实现方法还是有较大的差距（不是指效果），所以调用 setCardElevation也不能随心所欲地传入一个Float型，在低版本系统使用时应当处理一下传入的数值或加上try-catch（不推荐）。

**CardView属性介绍**

* android:cardCornerRadius
  + 在xml文件中设置card圆角的大小
* CardView.setRadius
  + 在代码中设置card圆角的大小
* android:cardBackgroundColor
  + 在xml文件中设置card背景颜色
* android:elevation
  + 在xml文件中设置阴影的大小
* card\_view:cardElevation
  + 在xml文件中设置阴影的大小
* card\_view:cardMaxElevation
  + 在xml文件中设置阴影最大高度
* card\_view:cardCornerRadius
  + 在xml文件中设置卡片的圆角大小
* card\_view:contentPadding
  + 在xml文件中设置卡片内容于边距的间隔
* card\_view:contentPaddingBottom
  + 在xml文件中设置卡片内容于下边距的间隔
* card\_view:contentPaddingTop
  + 在xml文件中设置卡片内容于上边距的间隔
* card\_view:contentPaddingLeft
  + 在xml文件中设置卡片内容于左边距的间隔
* card\_view:contentPaddingRight
  + 在xml文件中设置卡片内容于右边距的间隔
* card\_view:contentPaddingStart
  + 在xml文件中设置卡片内容于边距的间隔起始
* card\_view:contentPaddingEnd
  + 在xml文件中设置卡片内容于边距的间隔终止
* card\_view:cardUseCompatPadding
  + 在xml文件中设置内边距，V21+的版本和之前的版本仍旧具有一样的计算方式
* card\_view:cardPreventConrerOverlap
  + 在xml文件中设置内边距，在V20和之前的版本中添加内边距，这个属性为了防止内容和边角的重叠

**波纹点击效果**

* 默认情况，CardView是不可点击的，并且没有任何的触摸反馈效果。
* 触摸反馈动画在用户点击CardView时可以给用户以视觉上的反馈。
* 实现这种行为，你必须提供一下属性:android:clickable和android:foreground。
* <android.support.v7.widget.CardView
* xmlns:android="http://schemas.android.com/apk/res/android"
* xmlns:card\_view="http://schemas.android.com/apk/res-auto"
* ...
* android:clickable="true"
* android:foreground="?android:attr/selectableItemBackground">
* ...

</android.support.v7.widget.CardView>

**实例代码**

**引入的包**

dependencies {

compile fileTree(dir: 'libs', include: ['\*.jar'])

compile 'com.android.support:appcompat-v7:23.1.1'

compile 'com.android.support:cardview-v7:23.1.1'

compile 'com.android.support:recyclerview-v7:21.0.0'

compile 'com.nostra13.universalimageloader:universal-image-loader:1.9.3'

}

**MainActivity.java**

package com.cienet.android;

import android.app.Activity;

import android.os.Bundle;

import android.support.v7.widget.DefaultItemAnimator;

import android.support.v7.widget.LinearLayoutManager;

import android.support.v7.widget.RecyclerView;

import com.cienet.android.adapter.MyAdapter;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

public class MainActivity extends Activity {

private RecyclerView mRecyclerView;

private MyAdapter myAdapter;

private List<HashMap<String, String>> mList = new ArrayList<HashMap<String, String>>();

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

getPicUrls();

// 获取RecyclerView

mRecyclerView = (RecyclerView) findViewById(R.id.list);

// 设置LinearLayoutManager

mRecyclerView.setLayoutManager(new LinearLayoutManager(this));

// 设置ItemAnimator

mRecyclerView.setItemAnimator(new DefaultItemAnimator());

// 设置固定大小

mRecyclerView.setHasFixedSize(true);

// 初始化自定义的适配器

myAdapter = new MyAdapter(this, mList);

// 为mRecyclerView设置适配器

mRecyclerView.setAdapter(myAdapter);

}

private void getPicUrls() {

List<HashMap<String, String>> list = new ArrayList<HashMap<String, String>>();

HashMap<String, String> map1 = new HashMap<String, String>();

map1.put("A", "http://img3.hao123.com/data/1\_8583424a3f55c06ebeafce438a637c0d\_0");

list.add(map1);

HashMap<String, String> map2 = new HashMap<String, String>();

map2.put("A", "http://img0.hao123.com/data/1\_981ee6e65d3f13ec691804ab82f2a0ab\_510");

list.add(map2);

HashMap<String, String> map3 = new HashMap<String, String>();

map3.put("A", "http://img3.hao123.com/data/desktop/4926b79748d1c9d4db328e1a8b7a7794\_1280\_800");

list.add(map3);

HashMap<String, String> map4 = new HashMap<String, String>();

map4.put("A", "http://img5.hao123.com/data/1\_7793be4df6fd23d63ca321b205ba083b\_510");

list.add(map4);

HashMap<String, String> map5 = new HashMap<String, String>();

map5.put("A", "http://img3.hao123.com/data/1\_c46275cc6b24a480ecec31b3b5ec3c39\_510");

list.add(map5);

mList = list;

}

}

**MyAdapter.java**

package com.cienet.android.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.ImageView;

import com.cienet.android.R;

import com.nostra13.universalimageloader.core.ImageLoader;

import java.util.HashMap;

import java.util.List;

public class MyAdapter extends RecyclerView.Adapter<MyAdapter.ViewHolder> {

private Context mContext;

private List<HashMap<String, String>> mList;

public MyAdapter(Context context, List<HashMap<String, String>> list) {

this.mContext = context;

this.mList = list;

}

@Override

public ViewHolder onCreateViewHolder(ViewGroup viewGroup, int i) {

// 给ViewHolder设置布局文件

View v = LayoutInflater.from(viewGroup.getContext()).inflate(R.layout.card\_view, viewGroup, false);

return new ViewHolder(v);

}

@Override

public void onBindViewHolder(ViewHolder viewHolder, int i) {

// 给ViewHolder设置元素

ImageLoader.getInstance().displayImage(mList.get(i).get("A"), viewHolder.mImageView);

}

@Override

public int getItemCount() {

// 返回数据总数

return mList == null ? 0 : mList.size();

}

// 重写的自定义ViewHolder

public static class ViewHolder extends RecyclerView.ViewHolder {

public ImageView mImageView;

public ViewHolder(View v) {

super(v);

mImageView = (ImageView) v.findViewById(R.id.pic);

}

}

}

**card\_view.xml**

<android.support.v7.widget.CardView

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:card\_view="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal"

card\_view:cardBackgroundColor="@android:color/black"

card\_view:cardCornerRadius="10dp">

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="100dp"

android:padding="1dp">

<ImageView

android:id="@+id/pic"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_centerInParent="true"

android:scaleType="centerCrop" />

</RelativeLayout>

</android.support.v7.widget.CardView>

**main\_activity.xml**

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MyActivity">

<android.support.v7.widget.RecyclerView

android:id="@+id/list"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MyActivity" />

</FrameLayout>