ListView提高效率

（1）使用convertView

（2）使用持有者模式

BaseAdapter 抽象类

要继承BaseAdapter后进行实现适配器

activity\_main.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout 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=".MainActivity"**>  
  
 <**ListView  
 android:id="@+id/listView1"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentStart="true"  
 android:layout\_alignParentTop="true"** />  
</**RelativeLayout**>

list\_item.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="horizontal"**>  
  
 <**ImageView  
 android:id="@+id/imgName"  
 android:layout\_width="100dp"  
 android:layout\_height="100dp"  
 app:srcCompat="@drawable/a"** />  
  
 <**TextView  
 android:id="@+id/tvName"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="25dp"  
 android:text="TextView"  
 android:textAppearance="?android:attr/textAppearanceLarge"**/>  
</**LinearLayout**>

MainActivity.java

**package** com.amaker.hjs.adapterdemo3.baseadapter;  
  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.BaseAdapter;  
**import** android.widget.ImageView;  
**import** android.widget.ListView;  
**import** android.widget.TextView;  
  
**public class** MainActivity **extends** AppCompatActivity {  
 **private** ListView **listView**;  
 **private int**[] **imgResIds**={  
 R.drawable.***a***,  
 R.drawable.***b***,  
 R.drawable.***c***,  
 R.drawable.***d***,  
 R.drawable.***e***,  
 R.drawable.***f*** };  
 **private** String[] **names**={  
 **"evil"**,  
 **"woman"**,  
 **"love"**,  
 **"snow"**,  
 **"computer"**,  
 **"bird"** };  
  
 **private static int** *count*=0;*//测试数据* @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 **listView** = (ListView)findViewById(R.id.***listView1***);  
 **listView**.setAdapter(**new** MyAdapter());  
 }  
  
  
  
 **class** MyAdapter **extends** BaseAdapter{  
 */\*\*  
 \** ***@return*** *数据源的数据子项个数  
 \*/* @Override  
 **public int** getCount() {  
 **return** Integer.***MAX\_VALUE***;  
 }  
  
  
 */\*\*  
 \** ***@param position*** *数据子项的索引  
 \** ***@return*** *指定索引的数据子项  
 \*/* @Override  
 **public** Object getItem(**int** position) {  
 **return** position;  
 }  
  
  
 */\*\*  
 \** ***@param position*** *数据子项的索引  
 \** ***@return*** *指定索引的数据子项的id  
 \*/* @Override  
 **public long** getItemId(**int** position) {  
 **return** position;  
 }  
  
  
 */\*\*  
 \** ***@param position*** *数据子项的索引  
 \** ***@param convertView*** *dom解析，会把数据子项的xml文件载入到内存中，进行缓存。即如果某一索引项已经被解析过了，则该项会进行缓存，以便下一次直接使用提高效率  
 \** ***@param parent*** *指ListView  
 \** ***@return*** *指定索引的数据子项要显示的View  
 \*/* @Override  
 **public** View getView(**int** position, View convertView, ViewGroup parent) {  
 View view;  
 ViewHolder holder = **null**;  
 **if** (convertView==**null**){ *//使用convertView来提高效率  
 //count++;  
  
 //1.解析数据子项的布局文件，成为View* LayoutInflater inflater = LayoutInflater.*from*(MainActivity.**this**);*//如果是this，则意味着当前上下文是MyAdapter，所以必须指明为MainActivity.this* view = inflater.inflate(R.layout.***list\_item***,parent,**false**);  
  
 *//创建持有者* holder = **new** ViewHolder(view);  
  
 view.setTag(holder);*//在view上保存程序员所需的持有者数据（对象）* }**else** {  
 view = convertView;  
 holder= (ViewHolder) view.getTag();  
 }  
  
 *//2.填充数据到指定索引的数据子项  
  
 //ImageView image = (ImageView) view.findViewById(R.id.imgName);  
 //为了提高效率，减少findViewById，改为和view持有者拿这个ImageView控件* ImageView image = holder.getImage();  
 image.setImageResource(**imgResIds**[position%**names**.**length**]);  
 *//为了测试convertView，我通过getCount()方法指明了该数据子项有非常多个，而实际上我只有6个，  
 // 即[0,5]。因此我通过 模%6运算 来进行循环取值（position），并且能够防止数组下标越界  
  
  
 //TextView textView = (TextView)view.findViewById(R.id.tvName);  
 //为了提高效率，减少findViewById，改为和view持有者拿这个TextView控件* TextView textView = holder.getTextView();  
 textView.setText(**names**[position%**names**.**length**]+**"，count="**+*count*);  
 **return** view;  
 }  
 }  
  
  
 *//由于方法findViewById() 每次在内存中都需要去查找都非常耗费资源，我们可以设置一个持有者类，来持有当前类对象* **class** ViewHolder{  
 **private** View **view**;  
 **private** ImageView **image**;  
 **private** TextView **textView**;  
  
 **public** ViewHolder(View view){  
 **this**.**view** = view;  
 }  
  
 **public** ImageView getImage() {  
 **if**(**image**==**null**){  
 *//查找* **image** = (ImageView) **view**.findViewById(R.id.***imgName***);  
 }  
 **return image**;  
 }  
  
  
 **public** TextView getTextView() {  
 **if** (**textView**==**null**){  
 **textView**=(TextView) **view**.findViewById(R.id.***tvName***);  
 }  
 **return textView**;  
 }  
  
 }  
}





