ListView：在Android开发中ListView是比较常用的组件，它以列表的形式展示具体内容，并且能够根据数据的长度自适应显示

1. ListVeiw 用来展示列表的View。
2. 适配器 用来把数据映射到ListView上的中介。
3. 数据：具体的将被映射的字符串，图片，或者基本组件

案例1：采用普通的TextView来显示数据 (缺点不能滚动, 不能显示复制的数据). 技能点：如何把data显示到组件中

```
1.   private void showData() {
2.           // 模拟从数据库中获取PersonList对象
3.           perList.add(person);
4.           ...
5.           ...
6.     for(Person person:perList){
7.      TextView tv = new TextView(this);
8.      tv.setText(person.getId() + "," + person.getName() + ","
9.        + person.getSalary() + "," + person.getPhone());
10.     li.addView(tv);
11.    }
12.   }
```

给当前案例增加滚动的效果：(知道滚动布局嵌套其它布局,而且布局与布局之间可以相互嵌套)

```
1.   <ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
2.       xmlns:tools="http://schemas.android.com/tools"
3.       android:layout_width="match_parent"
4.       android:layout_height="match_parent" >
5.       <LinearLayout
6.           android:id="@+id/li"
7.           android:layout_width="match_parent"
8.           android:layout_height="wrap_content"
9.           android:orientation="vertical"
10.          android:paddingBottom="@dimen/activity_vertical_margin"
11.          android:paddingLeft="@dimen/activity_horizontal_margin"
12.          android:paddingRight="@dimen/activity_horizontal_margin"
13.          android:paddingTop="@dimen/activity_vertical_margin"
14.          tools:context=".MainActivity" >
15.      </LinearLayout>
16.
17.  </ScrollView>
```

如何解决(滚动+列表显示)问题. 采用ListView

- 首先在布局页面中添加ListView参数

```
1.   <ListView
2.          android:id="@+id/lv"
3.          android:layout_width="match_parent"
4.          android:layout_height="match_parent" >
5.   </ListView>
```

适配器设计模式：

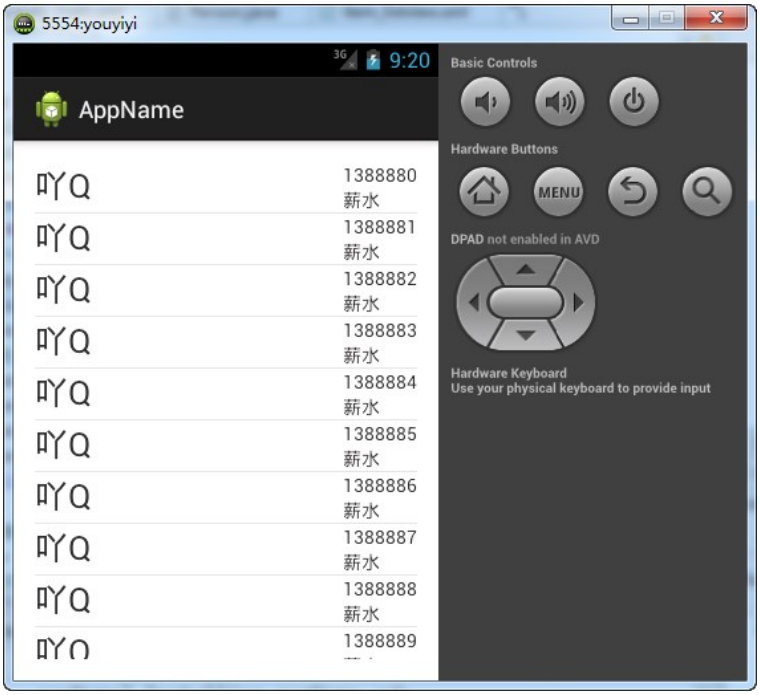
```
/*
 * 5.1.1 通过适配器，客户端可以调用同一接口，因而对客户端来说是透明的。这样做更简单、更直接、更紧凑。
 * 5.1.2 复用了现存的类，解决了现存类和复用环境要求不一致的问题。
 * 5.1.3 将目标类和适配者类解耦，通过引入一个适配器类重用现有的适配者类，而无需修改原有代码。
 * 5.1.4 一个对象适配器可以把多个不同的适配者类适配到同一个目标，也就是说，同一个适配器可以把适配者类和它的子类都适配到
 * */

// 系统期待的是Target接口
public class Adapter implements Target {

 private Adaptee adaptee = new Adaptee();

 @Override
 // 本质就是在Demo中调用了Test
 public void demo() {
  adaptee.test();
 }
}

// 需要适配的类型
class Adaptee {
 public void test() {
  System.out.println("---------test----------------");
 }
}
```

- 掌握适配器设计模式, 通过适配器模式,把相关的数据封装到具体的ViewItem中

```
private void showData() {
  // 模拟从数据库中获取PersonList对象
        final List<Person> perList = new ArrayList<Person>
         ...
         ...
  // 通过ID捆绑前端的ListView控件
  ListView lv = (ListView) findViewById(R.id.lv);
  lv.setAdapter(new MyAdapter());
 }

 private class MyAdapter extends BaseAdapter {

  @Override
  public int getCount() {
   // TODO Auto-generated method stub
   return perList.size();
  }

  @Override
  public View getView(int position, View convertView, ViewGroup parent) {
   TextView tv = new TextView(MainActivity.this);
   System.out.println("getView调用:" + position);
   Person person = perList.get(position);
```

```
21.       tv.setText(person.toString());
22.       tv.setTextSize(18);
23.       return tv;
24.     }
25.   }
```

优化ListView中的数据(需要在ListView中在部署几个子组件.) 结果如下:



首先自己创建一个布局文件：item_listview.xml

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3.      android:layout_width="match_parent"
4.      android:layout_height="match_parent" >
5.
6.      <TextView
7.          android:id="@+id/txt_name"
8.          android:layout_width="wrap_content"
9.          android:layout_height="wrap_content"
10.         android:text="名字"
11.         android:textSize="25sp" />
12.
13.     <LinearLayout
14.         android:layout_width="wrap_content"
15.         android:layout_height="wrap_content"
16.         android:layout_alignParentRight="true"
17.         android:orientation="vertical" >
18.
19.         <TextView
20.             android:id="@+id/txt_phone"
21.             android:layout_width="wrap_content"
22.             android:layout_height="wrap_content"
23.             android:text="号码" />
24.
25.         <TextView
26.             android:id="@+id/txt_salary"
```

```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="薪水" />
    </LinearLayout>

</RelativeLayout>
```

先把数据存储到item_listview中, 然后通过适配器设计模式把数据转化为ViewItem即可

```
@Override
public View getView(int position, View convertView, ViewGroup parent) {
 // 根据项目启动的代码： setContentView(R.layout.activity_main);findViewById(R.layout.item_listview) // 此方式默认是在activity_main中查找的
 // ViewGroup 默认是有子容器的,但是View默认是不能存储子容器的
 View v = View.inflate(MainActivity.this, R.layout.item_listview,null);
 // 注意是在v布局XML文件通过相应的id找到相应的组件
 TextView tv_name = (TextView) v.findViewById(R.id.txt_name);
 TextView tv_phone = (TextView) v.findViewById(R.id.txt_phone);
 TextView tv_salary = (TextView) v.findViewById(R.id.txt_salary);
    // 给数据赋值
 tv_name.setText(perList.get(position).getName());
 tv_phone.setText(perList.get(position).getPhone());
 tv_salary.setTag(perList.get(position).getSalary());
 return v;
}
```

关于View.inflate的解释说明：

一个Activity里如果直接用findViewByld(),对应的是setConentView()的那个layout里的组件，因此如果你的Activity里如果用到别的layout，你就必须用inflate()先将对话框上的layout找出来,然后再用这个layout对象去找到它上面的组件

```
@Override
protected void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 setContentView(R.layout.activity_main2);
}
```

```
View v = View.inflate(this,R.layout.activity_layout,null);
```

```
public View inflate(Context context, int Resourece,ViewGroup root)
  作用：填充一个新的视图层次结构从指定的XML资源文件中
  context : The Context object for your activity or application
  reSource：View的layout的ID
  root：  生成的层次结构的根视图
  return 填充的层次结构的根视图。如果参数root提供了，那么root就是根视图；否则填充的XML文件的根就是根视图。
```

优化 View组件代码

```
// 通过画图理解：convertView 仅仅是一个ViewList的缓存,里面的数据时需要动态加载的
if (convertView == null) {
 v = View.inflate(MainActivity.this, R.layout.item_listview,null);
} else {
```

```
    v = convertView;
  }
```

ArrayAdapter:是BaseAdapter的子类,如果只需要填充一种数据类型,则可以考虑使用它

```
  @Override
  protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);
   // 如果只显示一种类型,则课堂采用ArrayAdapter
   String[] obj=new String[]{"AAAA","BBBB","CCCC"};
   ListView lv=(ListView)super.findViewById(R.id.lv);
   lv.setAdapter(new ArrayAdapter<String>(this,
      R.layout.item_listview,R.id.tv_name,obj));
  }
```

在item_listview的xml文件中编写需要加载的控件数据

```
    <!-- 用来设置图片框架 -->
    <ImageView
        android:layout_width="40dp"
        android:layout_height="40dp"
        android:src="@drawable/ic_launcher" />

    <TextView
        android:id="@+id/tv_name"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textSize="22sp" />
```