**Настройка listview items с собственным ArrayAdapter**

[Pyatnitsev](https://pyatnitsev.ru/author/pyatnitsev/)[22.04.2013](https://pyatnitsev.ru/2013/04/22/costum-listview/)  [9 Comments](https://pyatnitsev.ru/2013/04/22/costum-listview/#comments)

 Просмотры: 17

Для реальных коммерческих мобильных приложений, стиль по-умолчанию и вид ListView в Android не привлекателен. Он может лишь отображать простую строку в каждой строке ListView, используя внутренний контролл (элемент) TextView. Для большинства приложений, вы захотите создать интерфейс, который более приятен для пользователя. Хорошо, что ListView очень мощный контрол и с помощью своих разметок элемента он может быть легко изменен что бы соответствовать вашим потребностям. В этом руководстве я покажу вам, как вы можете создать свои элементы ListView с иконками, своей разметкой заголовка и как вы можете  использовать настроенный ArrayAdapter что бы создавать что угодно на основе ArrayAdapter в будущем. Я также расскажу некоторые советы, которые вы можете использовать для оптимизации использования памяти вашим ListView.

[](http://pyatnitsev.ru/wp-content/uploads/2013/04/Android_Custom_ListView_ArrayAdapter.png)

Создайте новый проект Android в Eclipse с MainActivity, которая будет Activity-по умолчанию, а так же main.xml как разметка для этой Activity. Объявите элемент ListView в вашем файле разметки main.xml, как показано в следующем коде:

**Main.xml**

[Click to toggle codeblock](javascript:;" \o "Click to toggle codeblock)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | **<?xml** version="1.0" encoding="utf-8"**?>**  **<LinearLayout**     xmlns:android="http://schemas.android.com/apk/res/android"     android:orientation="vertical"     android:layout\_width="fill\_parent"     android:layout\_height="fill\_parent"     android:background="#FFFFFF"**>**    **<ListView**         android:id="@+id/listView1"         android:layout\_width="fill\_parent"         android:layout\_height="fill\_parent" **/>**    **</LinearLayout>** |

Код выше использует простой LinearLayout с вертикальной ориентацией и ListView объявляемый что бы покрыть всю ширину и высоту родительского контейнера используется fill\_parent как значение обоих *android:layout\_heightиandroid:layout:width* свойствах. ListView так же имеет уникальный id listView1, который будет использоваться в MainActivity что бы ссылаться на ListView.

Чтобы создать кастомизированный заголовок для ListView, создайте новый файл разметки xml -listview\_header\_row.xml в папке layout вашего проекта и объявите TextView его свойства показаны в следующем коде. Это создаст синий заголовок с белым текстом.

**listview\_header\_row.xml**

[Click to toggle codeblock](javascript:;)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | **<?xml** version="1.0" encoding="utf-8"**?>**  **<LinearLayout**     xmlns:android="http://schemas.android.com/apk/res/android"     android:orientation="horizontal"     android:layout\_width="fill\_parent"     android:layout\_height="fill\_parent"**>**    **<TextView** android:id="@+id/txtHeader"         android:layout\_width="fill\_parent"         android:layout\_height="fill\_parent"         android:gravity="center\_vertical"         android:layout\_alignParentTop="true"         android:layout\_alignParentBottom="true"         android:textStyle="bold"         android:textSize="22dp"         android:textColor="#FFFFFF"         android:padding="10dp"         android:text="Weather Photos"         android:background="#336699" **/>**    **</LinearLayout>** |

Что бы создать строку ListView, создайте другой файл разметки с именем listview\_item\_row.xml в папке layout вашего проекта. Android будет отрисовывать содержимое этого файла в каждом элементе ListView и вы можете определить любой контролл, который захотите. Для этого руководства я использую ImageView для иконки и TextView для отображения заголовков. Следующий код для listview\_item\_row.xml

**listview\_item\_row.xml**

[Click to toggle codeblock](javascript:;)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | **<?xml** version="1.0" encoding="utf-8"**?>**  **<LinearLayout**     xmlns:android="http://schemas.android.com/apk/res/android"     android:orientation="horizontal"     android:layout\_width="fill\_parent"     android:layout\_height="fill\_parent"     android:padding="10dp"**>**    **<ImageView** android:id="@+id/imgIcon"         android:layout\_width="wrap\_content"         android:layout\_height="fill\_parent"         android:gravity="center\_vertical"         android:layout\_alignParentTop="true"         android:layout\_alignParentBottom="true"         android:layout\_marginRight="15dp"         android:layout\_marginTop="5dp"         android:layout\_marginBottom="5dp" **/>**    **<TextView** android:id="@+id/txtTitle"         android:layout\_width="fill\_parent"         android:layout\_height="fill\_parent"         android:gravity="center\_vertical"         android:layout\_alignParentTop="true"         android:layout\_alignParentBottom="true"         android:textStyle="bold"         android:textSize="22dp"         android:textColor="#000000"         android:layout\_marginTop="5dp"         android:layout\_marginBottom="5dp" **/>**    **</LinearLayout>** |

Для этого руководства я скачал некоторые иконки 32 x 32 в формате PNG. Вы можете использовать ваши собственные иконки по своему усмотрению. Когда иконки готовы перетащите иконки из вашей директории (папки) в папку  drawable-mdpi внутри вашего проекта. Далее создайте новый класс JAVA в вашем проекте с именем **Weather.java.**Этот класс будет использоваться кастомизированным ArrayAdapter и будет позже связывать объекты с ListView. Следующий код для  **Weather.java.** В нем есть два простых свойства: иконка и заголовок, а так же простой конструктор класса, что бы инициализировать свойства.

**Weather.java**

[Click to toggle codeblock](javascript:;" \o "Click to toggle codeblock)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | public class Weather {      public int icon;      public String title;      public Weather(){          super();      }        public Weather(int icon, String title) {          super();          this.icon = icon;          this.title = title;      }  } |

Заметьте, что выше  файл listview\_item\_row.xml имеет 2 view, которые передают свойства класса Weather. Значения свойств класса Weather будут отображаться на их view а так же соениять эти 2 места вместе вы должны создать кастомный ArrayAdapter, который будет наследоваться от класса  Android ArrayAdapter и будет переопределять метод getView. Добавьте java-класс в ваш проект под именем nameWeatherAdapter и реализуйте код, который показан ниже.

**WeatherAdapter.java**

[Click to toggle codeblock](javascript:;" \o "Click to toggle codeblock)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47 | public class WeatherAdapter extends ArrayAdapter**<Weather>**{        Context context;      int layoutResourceId;      Weather data[] = null;        public WeatherAdapter(Context context, int layoutResourceId, Weather[] data) {          super(context, layoutResourceId, data);          this.layoutResourceId = layoutResourceId;          this.context = context;          this.data = data;      }        @Override      public View getView(int position, View convertView, ViewGroup parent) {          View row = convertView;          WeatherHolder holder = null;            if(row == null)          {              LayoutInflater inflater = ((Activity)context).getLayoutInflater();              row = inflater.inflate(layoutResourceId, parent, false);                holder = new WeatherHolder();              holder.imgIcon = (ImageView)row.findViewById(R.id.imgIcon);              holder.txtTitle = (TextView)row.findViewById(R.id.txtTitle);                row.setTag(holder);          }          else          {              holder = (WeatherHolder)row.getTag();          }            Weather weather = data[position];          holder.txtTitle.setText(weather.title);          holder.imgIcon.setImageResource(weather.icon);            return row;      }        static class WeatherHolder      {          ImageView imgIcon;          TextView txtTitle;      }  } |

В коде выше первая важная вещь — это конструктор класса, у которого есть три параметра. Первый параметр это Context и мы можем ссылаться на activity в которой мы будем использовать класс WeatherAdapter. Второй параметр это идентификатор ресурса файла разметки, который мы хотим использовать что бы отображать каждый элемент в ListView. Мы будем получать идентификатор ресурса файла разметки listview\_item\_row.xml в этом параметре. Третий параметр — это массив экземпляров класса Weather, который будет использоваться адаптером, для отображения данных.

Метод getView из родительского класса переопределяется. Этот метод будет вызываться для каждого элемента в ListView что бы создать представления с их свойствами, как мы хотим. Метод getView также используется как временное хранилище класса определенного внутри класса  WeatherAdapter. Этот класс будет использовать что бы кешировать ImageView и TextView, они могут быть использованы снова для каждой строки в ListView и он будет обеспечивать нас хорошим увеличением производительности, так как мы можем переопределить некоторые два представления с разными свойствами и нам не нужно искать элементы ImageView и TextView для каждого элемента списка ListView. Код выше также используется Android built в Layout Inflator что бы обрабатывать файл разметки.

Последняя часть кода это MainActivity который будет использовать объекты, объявленные выше. Следующий код из файла **MainActivity.java**

**MainActivity.java**

[Click to toggle codeblock](javascript:;)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | public class MainActivity extends Activity {        private ListView listView1;        @Override      public void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.main);            Weather weather\_data[] = new Weather[]          {              new Weather(R.drawable.weather\_cloudy, "Cloudy"),              new Weather(R.drawable.weather\_showers, "Showers"),              new Weather(R.drawable.weather\_snow, "Snow"),              new Weather(R.drawable.weather\_storm, "Storm"),              new Weather(R.drawable.weather\_sunny, "Sunny")          };            WeatherAdapter adapter = new WeatherAdapter(this,                  R.layout.listview\_item\_row, weather\_data);            listView1 = (ListView)findViewById(R.id.listView1);            View header = (View)getLayoutInflater().inflate(R.layout.listview\_header\_row, null);          listView1.addHeaderView(header);            listView1.setAdapter(adapter);      } |

Несколько вещей в MainActivity которые требуются для объяснения для вашего лучшего понимания. Первое замечание, мы создаем массив экземпляров класса Weather, а так же иконок и заголовков которые передаются в конструктор в качестве параметров. Далее, объект WeatherAdapter создан и файл разметки listview\_item\_row.xml id и объекты класса Weather в массиве передаются в конструктор. Еще один раз, мы используем Android Layout Inflator что бы заполнить файл разметки нашгего listview\_header\_row.xml, это обрабатывается как заголовок для View. Это будет передано как параметр метода addHeaderView.

В конце мы передаем наш кастомный адаптер в метод setAdapter.

На этом мы готовы скомпилировать и запустить наш проект. Если все реализовано правильно, вы увидите следующий вывод:

[](http://pyatnitsev.ru/wp-content/uploads/2013/04/Android_Custom_ListView_ArrayAdapter.png)

Эта статья является [переводом отсюда](http://www.ezzylearning.com/tutorial.aspx?tid=1763429)