



# آموزش برنامه نویسی اندروید در محیط اندروید استودیو

**کار با ListView**

مدرس : سید مهدی مطهری

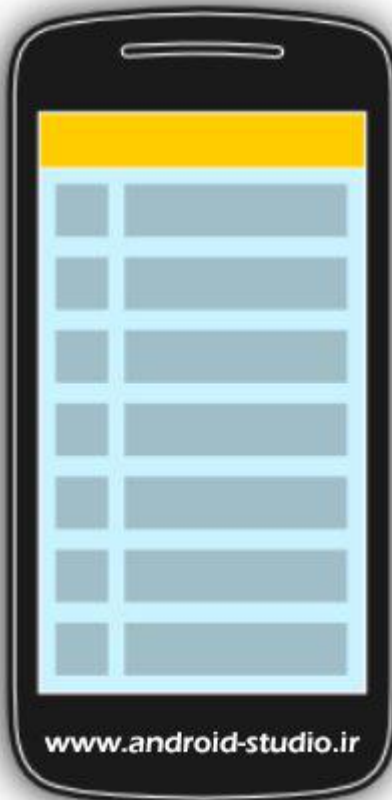
[www.android-studio.ir](http://www.android-studio.ir)



## به نام خدا

### ListView چیست

ListView یکی از ویجت های پرکاربرد در اپلیکیشن های اندروید است. همانطور که از نام آن پیداست، برای نمایش لیست بکار می رود.



در واقع ListView لیستی از آیتم ها را نمایش می دهد که قابل Scroll بوده و محدودیتی در تعداد آیتم ها ندارد. از نمونه های کاربردی می توان به لیست مخاطبین تلفن همراه، لیست ایمیل ها و یا لیست شهرها به همراه اطلاعات آب و هوایی مربوط به آنها در یک اپلیکیشن هواشناسی اشاره کرد.



در `ListView` امکان شخصی سازی وجود دارد و آیتم ها از ساده ترین حالت یعنی یک `text` تا یک آیتم پیچیده (ترمیمی از تصویر، متن، لینک و...) قابل پیاده سازی هستند. داده ها از منابعی مانند آرایه ها و دیتابیس به `ListView` منتقل و نمایش داده می شوند.

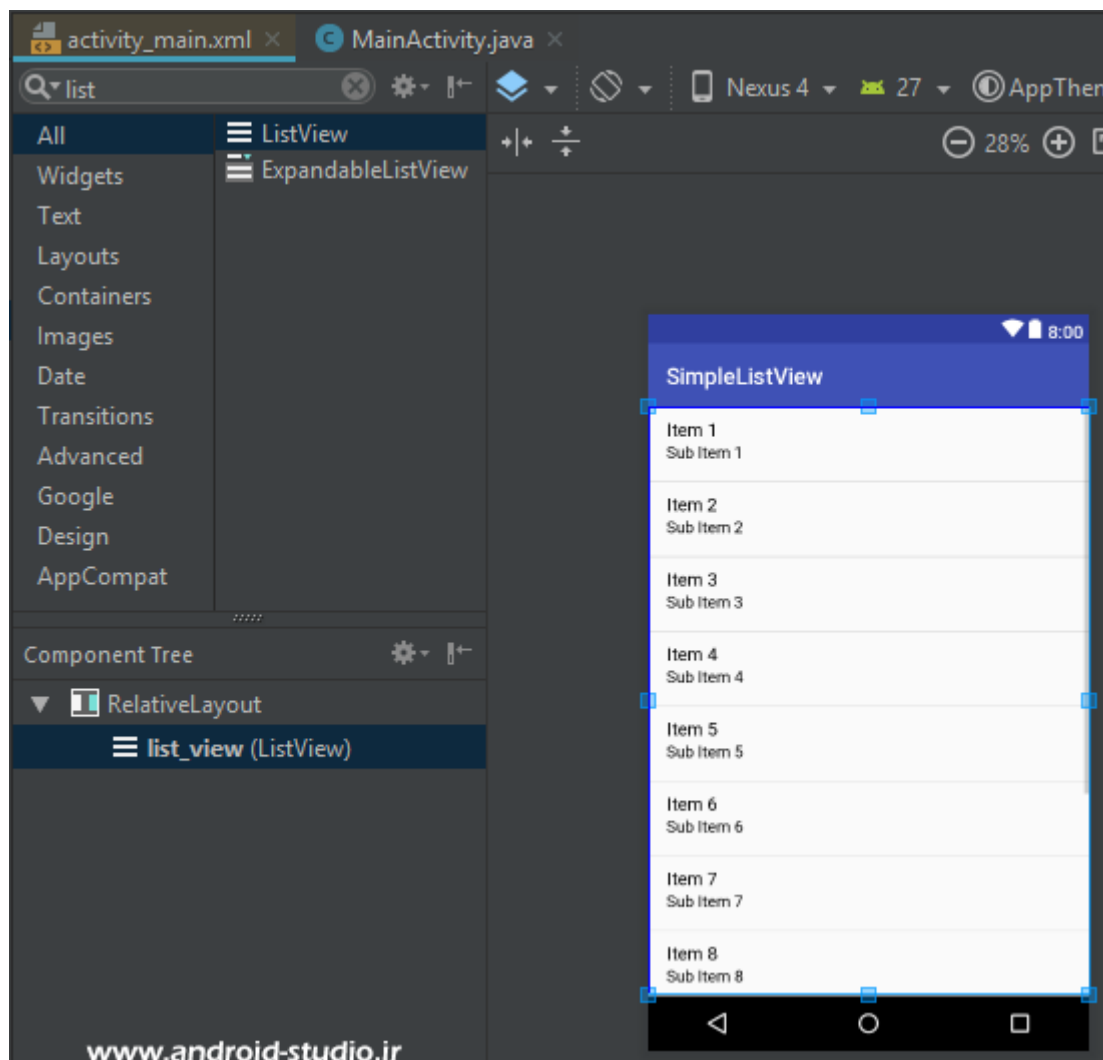
البته در متریال دیزاین یک جایگزین با نام `RecyclerView` برای `ListView` معرفی شده که مدیریت آن ساده تر بوده و عملکرد بهتری نیز نسبت به `ListView` دارد. با این حال لازم دانستم این ویجت را در حد ضرورت معرفی کنم.

در این مبحث، `ListView` در قالب دو پروژه مجزا بررسی می شود که پروژه اول یک `ListView` ساده و پروژه دوم یک `Custom ListView` (سفارشی) است.



## 1- ListView ساده

یک پروژه با نام SimpleListView و یک Empty Activity در اندروید استودیو ایجاد می‌کنم. ابتدا لازم است یک ویجت از نوع ListView به Layout اکتیویتی اضافه کنم که علاوه بر اضافه کردن دستی در محیط Editor، در محیط Design نیز به راحتی با Drag & Drop کردن ListView از Palette به روی صفحه Preview (یا قسمت Component Tree و در زیر RelativeLayout) به لایه اضافه می‌شود. سپس برای ویجت یک ID تعریف می‌کنم. پس از تعریف ID، یک لیست نمایشی به صفحه Preview اضافه می‌شود که صرفاً نمایشی است و فعلاً با اجرا روی دیوایس یا شبیه ساز، لیستی نمایش داده نمی‌شود.





## :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="ir.android_studio.simplelistview.MainActivity">

    <ListView
        android:id="@+id/list_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true" />

</RelativeLayout>
```

سپس ویجت ListView را مانند سایر ویجت هایی که قبلا کار کردیم، در اکتیویتی تعریف می کنیم:

## :MainActivity.java

```
package ir.android_studio.simplelistview;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ListView;

public class MainActivity extends AppCompatActivity {

    ListView mListview;

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

        mListview = findViewById(R.id.list_view);
    }
}
```

در ادامه یک آرایه تعریف می کنیم و سپس به واسطه Adapter مقادیر آرایه را به ListView انتقال می دهیم:



```

package ir.android_studio.simplelistview;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;

public class MainActivity extends AppCompatActivity {

    ListView mListview;
    String[] mobileBrands;

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

        mListview = findViewById(R.id.list_view);

        mobileBrands = new String[] {"LG", "Samsung", "Apple", "Sony", "Huawei",
            "HTC", "Motorola"};

        ArrayAdapter<String> mAdapter = new ArrayAdapter<String>(this,
            android.R.layout.simple_list_item_1, mobileBrands);

        mListview.setAdapter(mAdapter);
    }
}

```

به کد بالا دقت کنید. ابتدا یک آرایه از جنس `String[]` با نام `mobileBrands` ساخته و چند مقدار شامل نام برندها را در آرایه قرار داده ام. سپس برای انتقال و نمایش آرایه ها در ویجت `ListView` به یک `Adapter` نیاز داریم. یک نمونه با نام دلخواه `mAdapter` از کلاس `ArrayAdapter` و نوع `String` ایجاد کردم که سه پارامتر ورودی نیاز دارد:

```

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

    mListview = findViewById(R.id.list_view);

    mobileBrands = new String[] {"LG", "Samsung", "Apple", "Sony", "Huawei", "HTC", "Motorola"};

    ArrayAdapter<String> mAdapter = new ArrayAdapter<String>();
}

```

**@NonNull Context context**, @LayoutRes int resource  
**@NonNull Context context**, @LayoutRes int resource, @IdRes int textViewResourceid  
**@NonNull Context context**, @LayoutRes int resource, @NonNull String[] objects  
**@NonNull Context context**, @LayoutRes int resource, @IdRes int textViewResourceid, @NonNull String[] objects  
**@NonNull Context context**, @LayoutRes int resource, @NonNull List<String> objects  
**@NonNull Context context**, @LayoutRes int resource, @IdRes int textViewResourceid, @NonNull List<String> objects



پارامتر اول Context است که مطابق گذشته this وارد می‌کنم. چون تصمیم دارم لیست در همین اکتیویتی اجرا شود. پارامتر بعدی layout ای است که داده‌ها در قالب آن به ترتیب و زیر هم به ListView فرستاده می‌شوند. من فعلا ساده‌ترین کار را انجام می‌دهم. یعنی استفاده از layout پیش فرض اندروید با نام simple\_list\_item\_1:

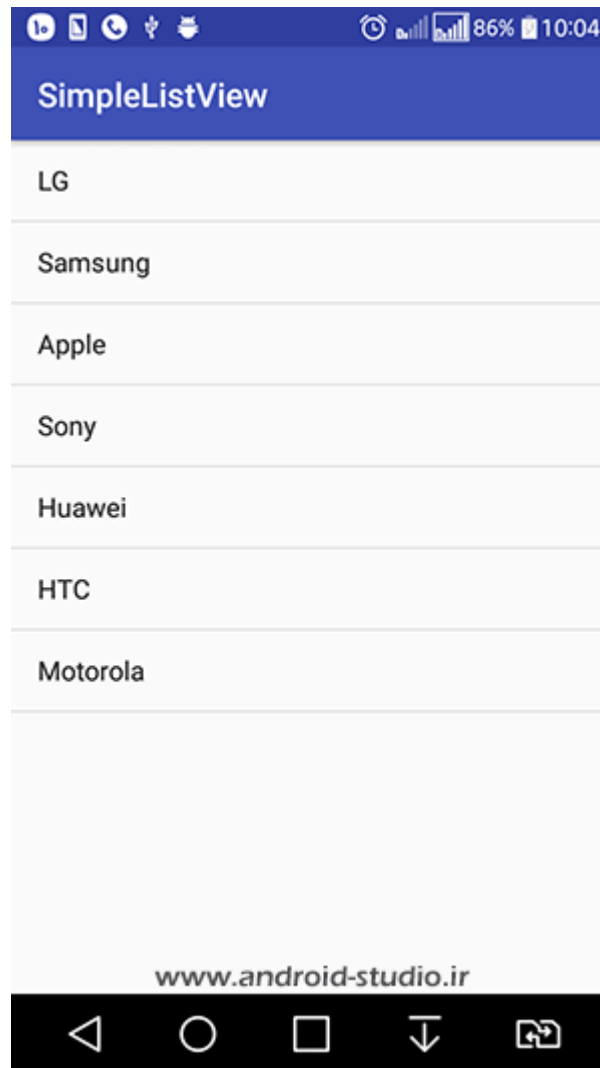
```
new ArrayAdapter<String>( context: this, android.R.layout.);
```

simple_list_item_1	( = 17367043)	int
simple_list_item_2	( = 17367044)	int
activity_list_item	( = 17367040)	int
browser_link_context_header	( = 17367054)	int
expandable_list_content	( = 17367041)	int
list_content	( = 17367060)	int
preference_category	( = 17367042)	int
select_dialog_item	( = 17367057)	int
select_dialog_multichoice	( = 17367059)	int
select dialog singlechoice	( = 17367058)	int

www.android-studio.ir

پارامتر سوم هم مربوط به نام آرایه می‌شود. یعنی mobileBrands.

Adapter ما کامل شد. در مرحله نهایی توسط متد setAdapter داده‌ها از mAdapter به ListView ارسال می‌شود. پروژه را اجرا می‌کنم:



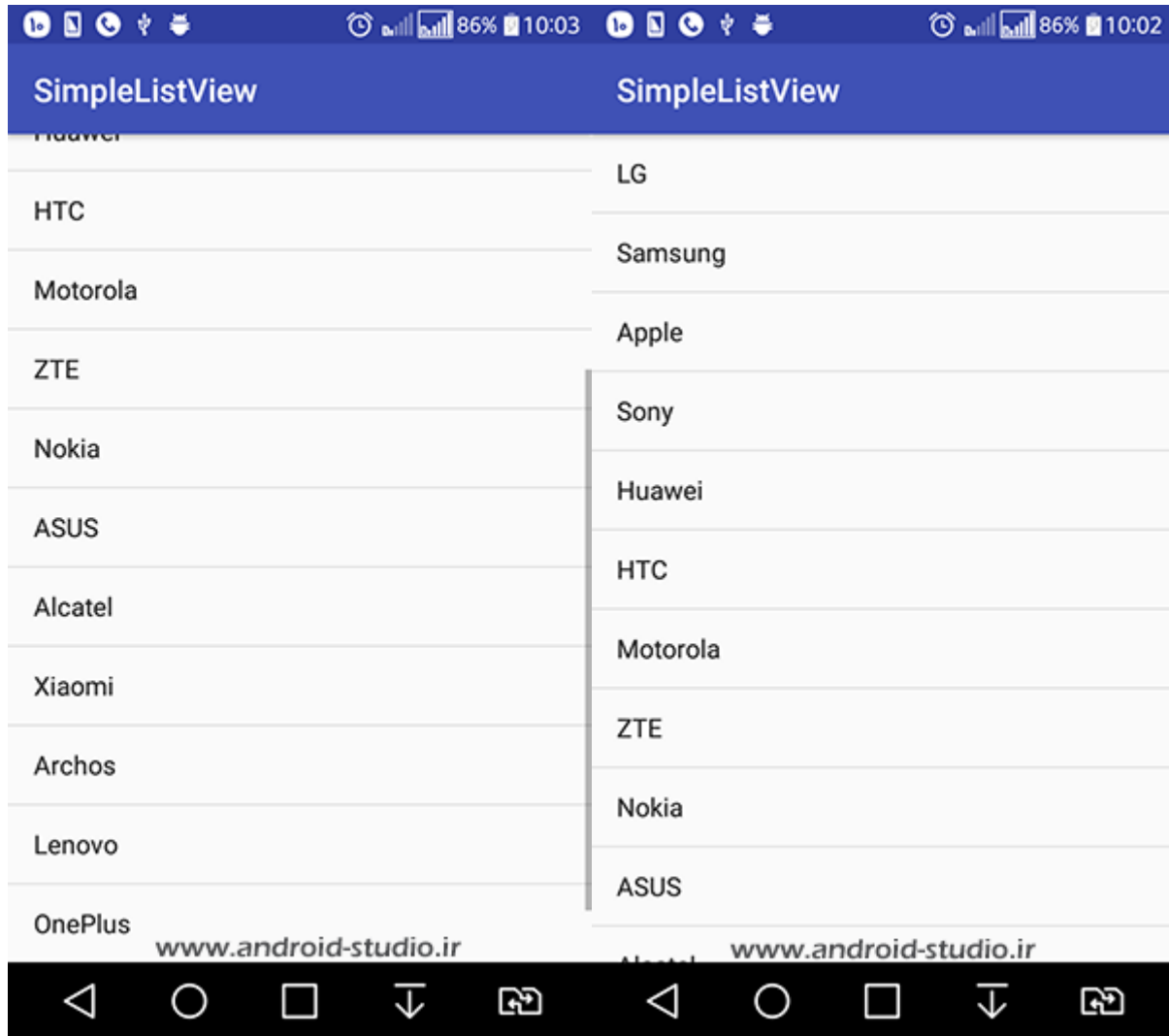
مشاهده می کنید لیست به درستی ساخته شده و داده های موجود در آرایه به ساده ترین شکل ممکن زیر یکدیگر قرار گرفته اند.

چند داده دیگر به آرایه اضافه می کنم تا اسکرول شدن لیست را نشان دهم:

```
mobileBrands = new String[] {"LG", "Samsung", "Apple", "Sony", "Huawei", "HTC", "Motorola",
    "ZTE", "Nokia", "ASUS", "Alcatel", "Xiaomi", "Archos", "Lenovo", "OnePlus", "OPPO"};
```

با اجرای مجدد پروژه، آیتم های جدید به لیست اضافه شده که با اسکرول کردن صفحه، به همه آیتم ها دسترسی داریم:





کلید Ctrl را نگه داشته و روی simple\_list\_item\_1 کلیک کنید:

```
mobileBrands = new String[] {"LG", "Samsung", "Apple", "Sony", "Huawei", "HTC", "Motorola", "ZTE", "OPPO", "Nokia", "ASUS", "Alcatel", "Xiaomi", "Archos", "Lenovo", "OnePlus", "OPPO"};
// Simple List Item:
ArrayAdapter<String> mAdapter = new ArrayAdapter<String>( context this, android.R.layout.simple_list_item_1, mobileBrands);
```



فایل simple\_list\_item\_1.xml در تب جدید باز می شود:

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Copyright (C) 2006 The Android Open Source Project

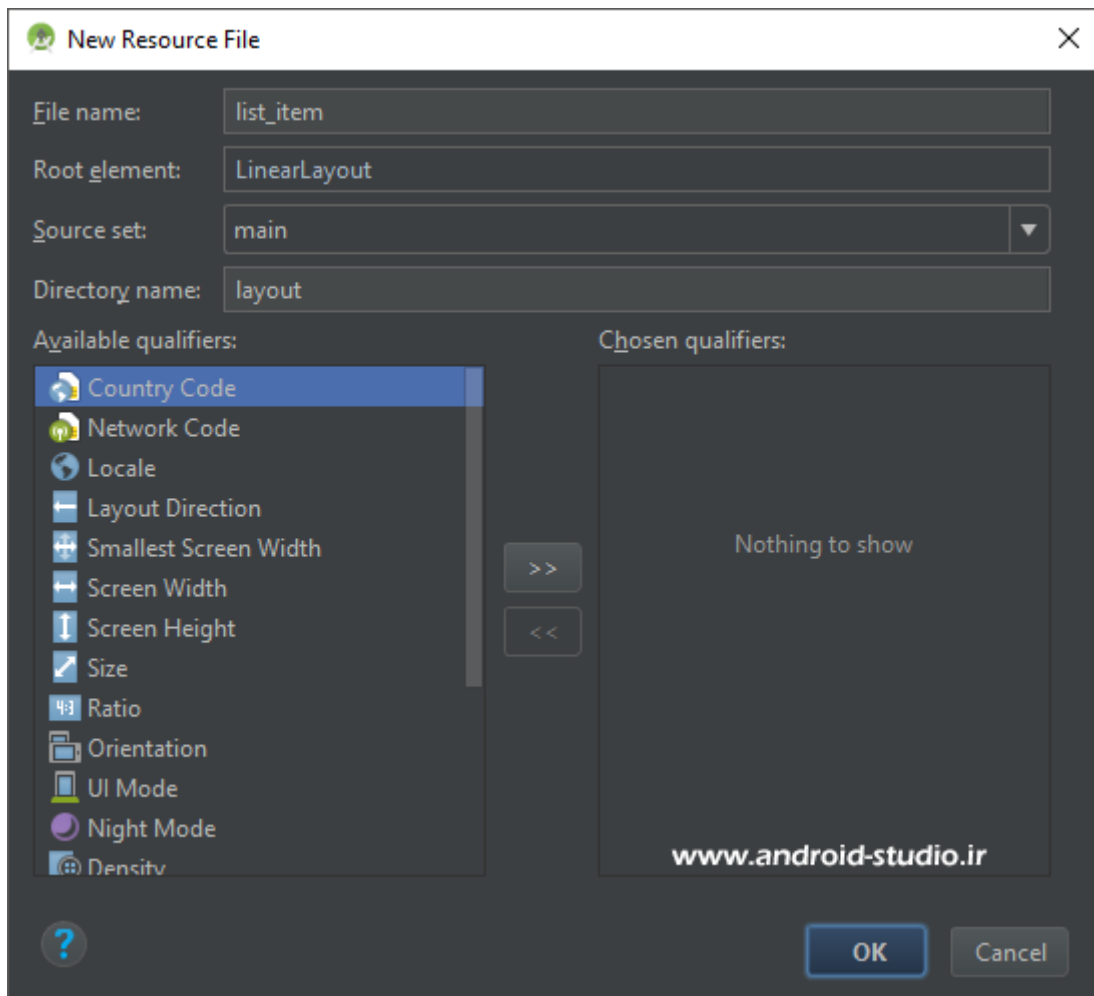
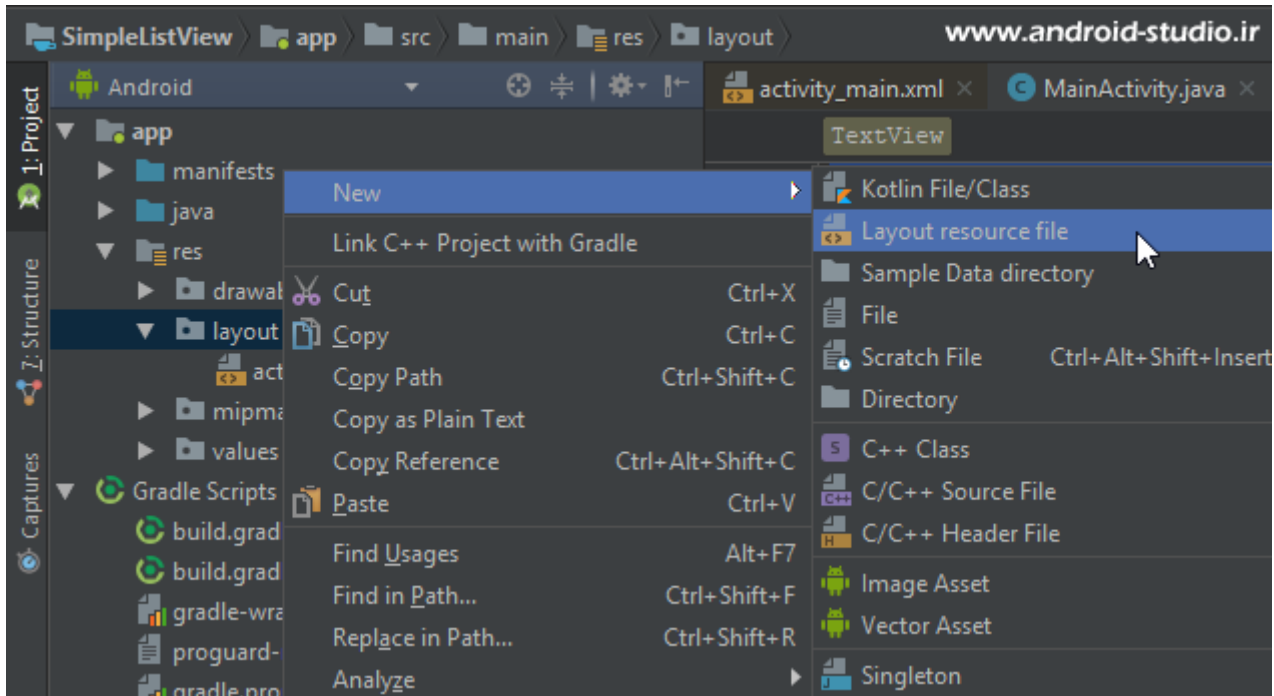
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
-->
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@android:id/text1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceListItemSmall"
    android:gravity="center_vertical"
    android:paddingStart="?android:attr/listPreferredItemPaddingStart"
    android:paddingEnd="?android:attr/listPreferredItemPaddingEnd"
    android:minHeight="?android:attr/listPreferredItemHeightSmall" />
```

مشاهده می کنید فقط یک TextView درون این فایل قرار داده شده با تعدادی خاصیت جهت استایل دهی نحوه نمایش آیتم ها. Layout ای که در ArrayAdapter به عنوان دومین پارامتر ورودی معرفی می کنیم، برای یک آیتم طراحی شده که Adapter تک تک داده ها را در این layout تکرار کرده و به ListView می فرستند که نتیجه اش همان چیزی می شود که با اجرای پروژه می بینیم. یعنی یک لیست از داده ها. جهت درک ساده تر این موضوع، به جای استفاده از لایه ی پیش فرض اندروید، یک لایه به پروژه اضافه می کنم:

راست کلیک روی فولدر layout و گزینه layout resource file





یک لایه جدید با نام دلخواه list\_item.xml به پروژه و فولدر layout اضافه شد. یک TextView به لایه اضافه می‌کنم:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/text_view"/>

</LinearLayout>
```

حالا در ArrayAdapter لایه جدید را جایگزین لایه پیش فرض می‌کنم:

```
ArrayAdapter<String> mAdapter = new ArrayAdapter<String>(this, R.layout.list_item,
R.id.text_view, mobileBrands );
```

layout قبل از لایه‌های موجود در SDK اندروید بود که با android.R.layout قابل دسترسی است اما لایه جدید درون پروژه ما قرار دارد بنابراین با R.layout تعریف می‌شود. در اینجا یک پارامتر به ArrayAdapter اضافه می‌شود که مربوط به آی دی TextView است که من آی دی text\_view را برای ویجت TextView در نظر گرفته بودم.

پروژه را اجرا می‌کنم:

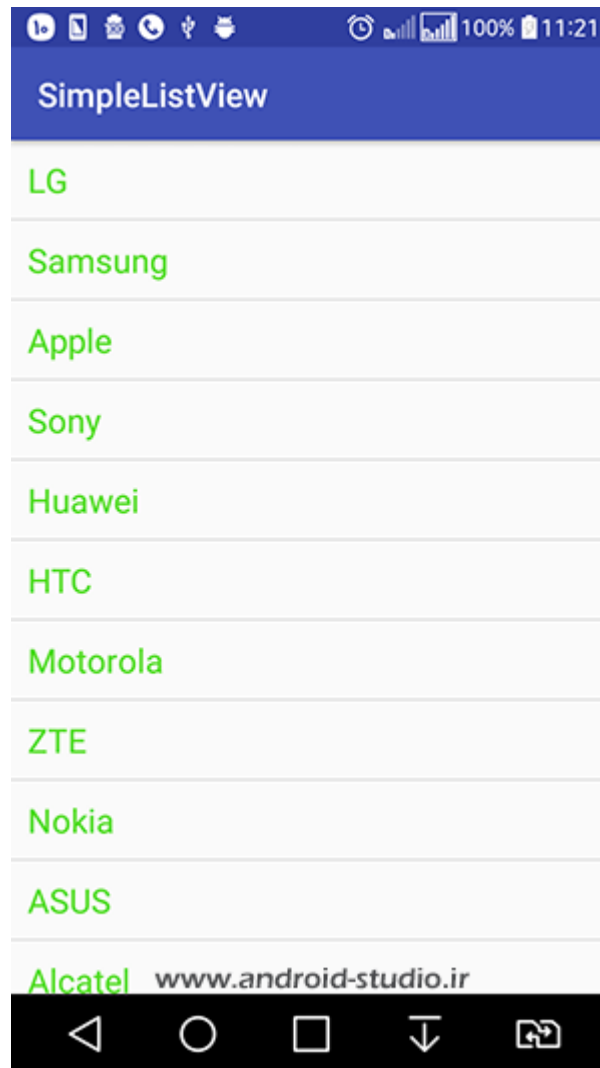


حالا TextView را به سلیقه خود می توانیم تغییر دهیم:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/text_view"
        android:textColor="#4edc1f"
        android:padding="10dp"
        android:textSize="20sp"/>

</LinearLayout>
```

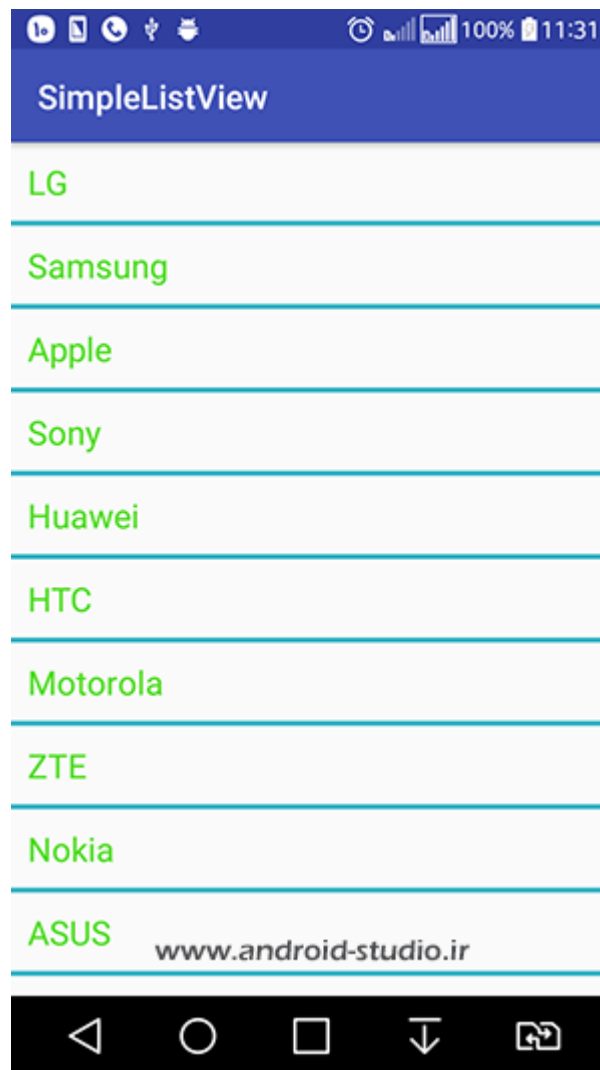


برای ویجت ListView هم چندین خاصیت معرفی شده. به عنوان مثال با استفاده از divider و dividerHeight می‌توان رنگ و ارتفاع خط جداکننده آیتم‌ها را تعیین کرد:

```
<?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="ir.android_studio.simplelistview.MainActivity">

    <ListView
        android:id="@+id/list_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:divider="#36b3c4"
        android:dividerHeight="3dp"/>

</RelativeLayout>
```



خاصیت دیگری برای این ویجت با نام `entries` معرفی شده است. تا اینجا ما آرایه را از یک کلاس جاوا به لایه رابط کاربری منتقل کردیم. اما راه دیگری هم برای تعریف آرایه وجود دارد که ممکن است در برخی مواقع گزینه مناسبتر و ساده تری نسبت به استفاده از جاوا باشد. با استفاده از خاصیت `entries` این امکان فراهم می شود تا بتوانیم آرایه را از یک فایل `xml` دریافت کنیم.

ابتدا خطهای اضافی در `MainActivity.java` را حذف می کنیم تا فقط کد مربوط به اتصال `ListView` به اکتیویته باقی بماند:



```
package ir.android_studio.simplelistview;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ListView;

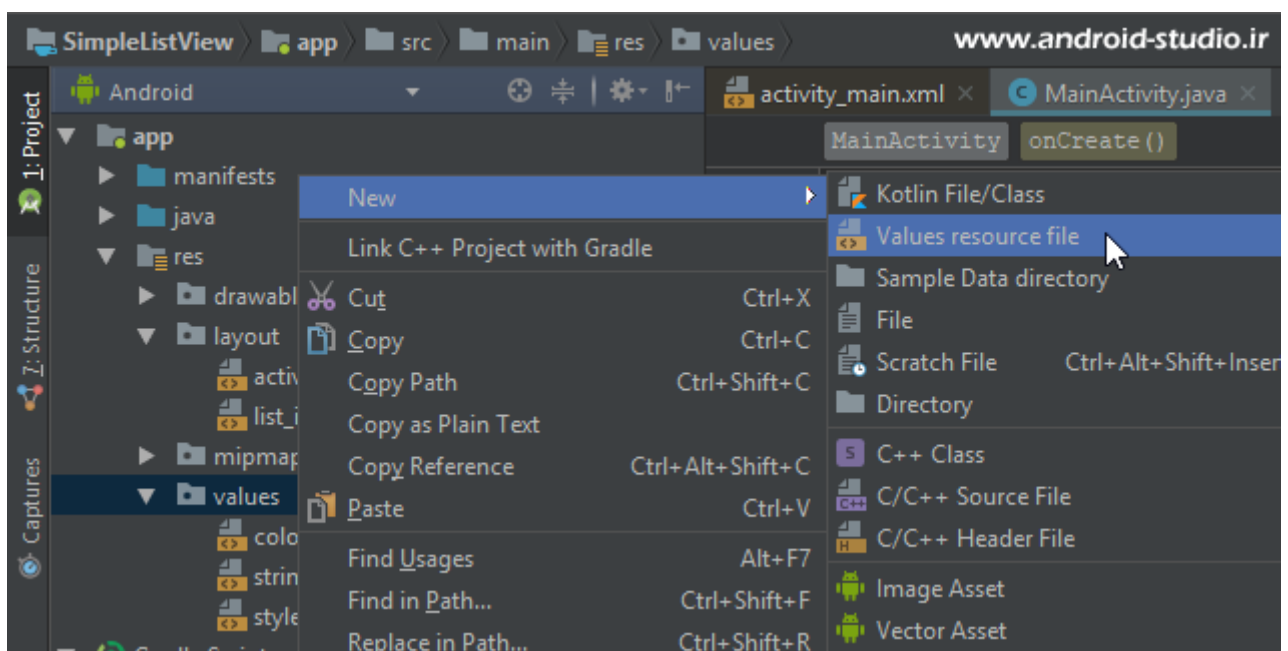
public class MainActivity extends AppCompatActivity {

    ListView mListView;

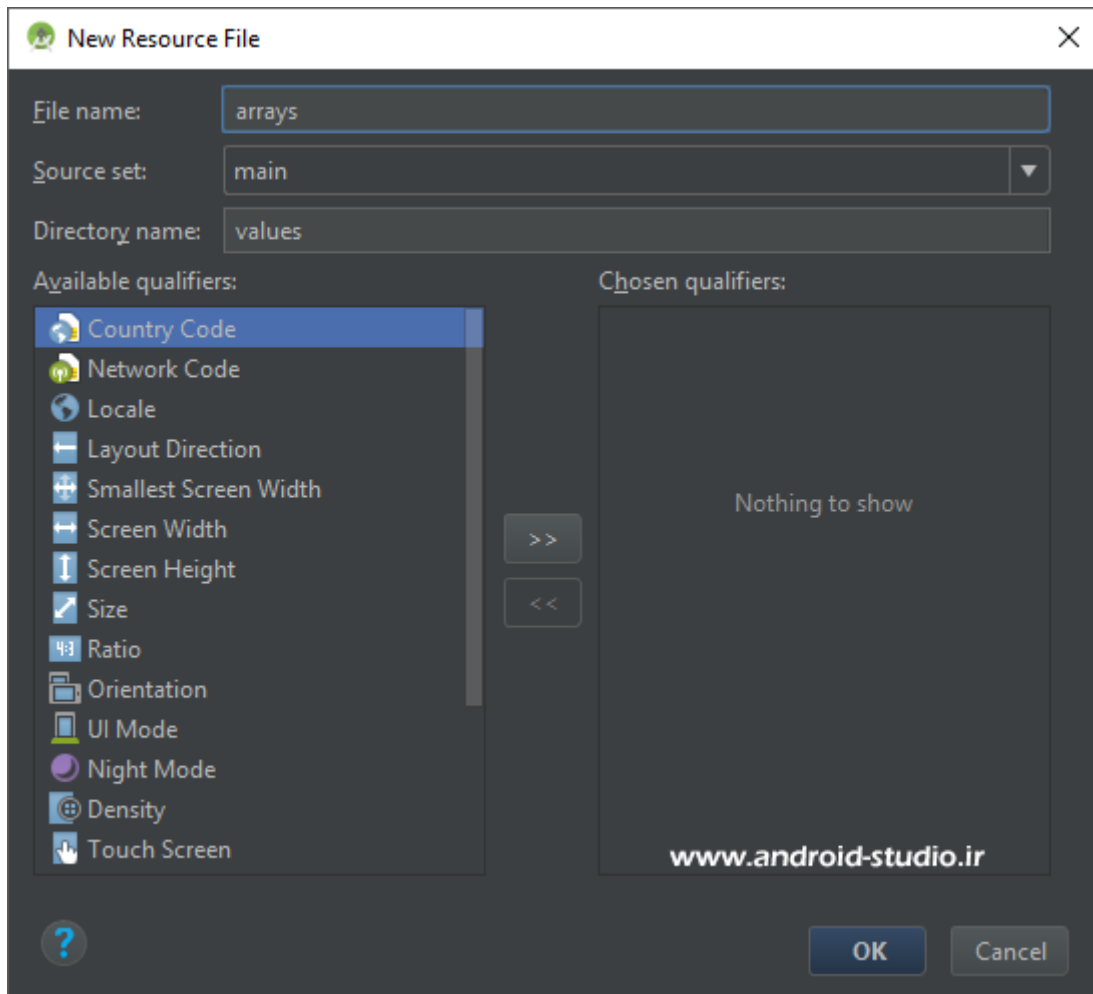
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mListView = findViewById(R.id.list_view);
    }
}
```

سپس یک فایل با نام arrays.xml به دایرکتوری values پروژه اضافه می‌کنم:







داخل arrays.xml یک تگ array با نام دلخواه names ساخته و تعدادی آیتم به آن اضافه نمودم:

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

  <array name="names">
    <item>Mahdi</item>
    <item>Mostafa</item>
    <item>Zahra</item>
    <item>Fateme</item>
    <item>Arman</item>
  </array>

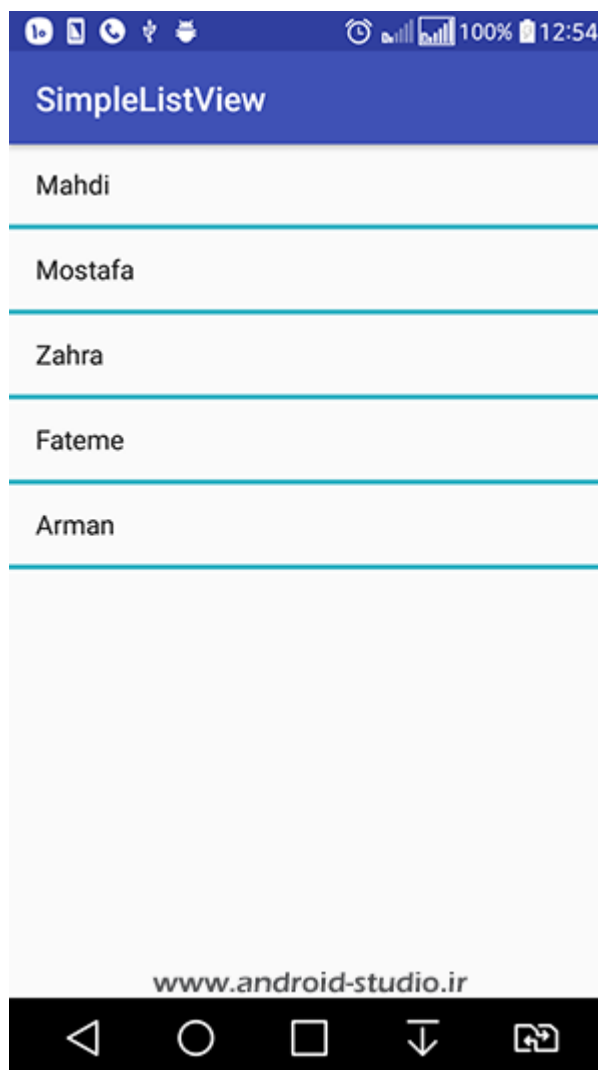
</resources>
```



حالا کفایت به واسطه خاصیت entries این آرایه را به ListView متصل کنیم:

```
<ListView
    android:id="@+id/list_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_alignParentStart="true"
    android:layout_alignParentTop="true"
    android:divider="#36b3c4"
    android:dividerHeight="3dp"
    android:entries="@array/names"/>
```

با اجرای پروژه آرایه هایی که در xml تعریف کردیم، در صفحه نمایش داده می شود:



**نکته:** در حالت استفاده از xml برای تعریف آرایه، داده ها مستقیم به ListView منتقل می شوند بنابراین فایل list\_item.xml که قبلا ساختیم در اینجا نقشی ندارد.



## ۲- ListView سفارشی

در پروژه قبل یک ListView بسیار ساده ساختیم که فقط یک String را از آرایه دریافت می کرد. در این مرحله یک Custom ListView می سازیم که شامل یک ImageView و چند TextView است.

یک پروژه با نام CustomListView و یک Empty Activity ایجاد می کنم.

مانند پروژه قبل ابتدا یک ویجت از جنس ListView به لایه اکتیویتی اضافه می کنم:

: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="ir.android_studio.customlistview.MainActivity">

    <ListView
        android:id="@+id/list_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

</RelativeLayout>
```

سپس یک layout جدید با نام دلخواه list\_item.xml به پروژه اضافه کرده و ویجت های مدنظر را در آن قرار می دهیم:



```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/layout_root"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <ImageView
        android:id="@+id/img_avatar"
        android:layout_width="wrap_content"
        android:layout_height="80dp"
        android:src="@drawable/avatar_1" />

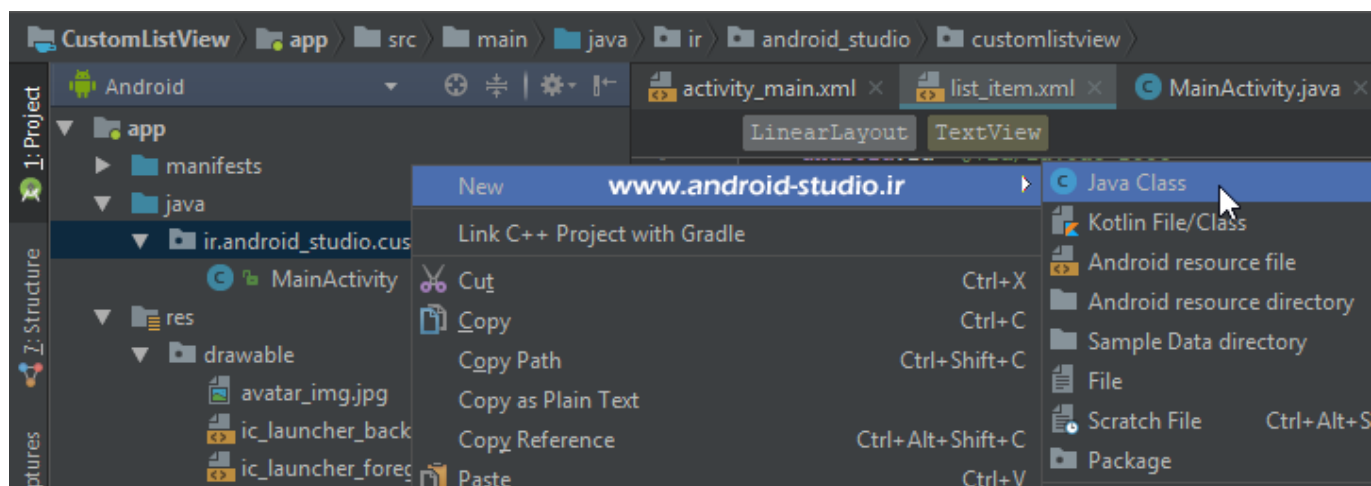
    <TextView
        android:id="@+id/txt_name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Name" />

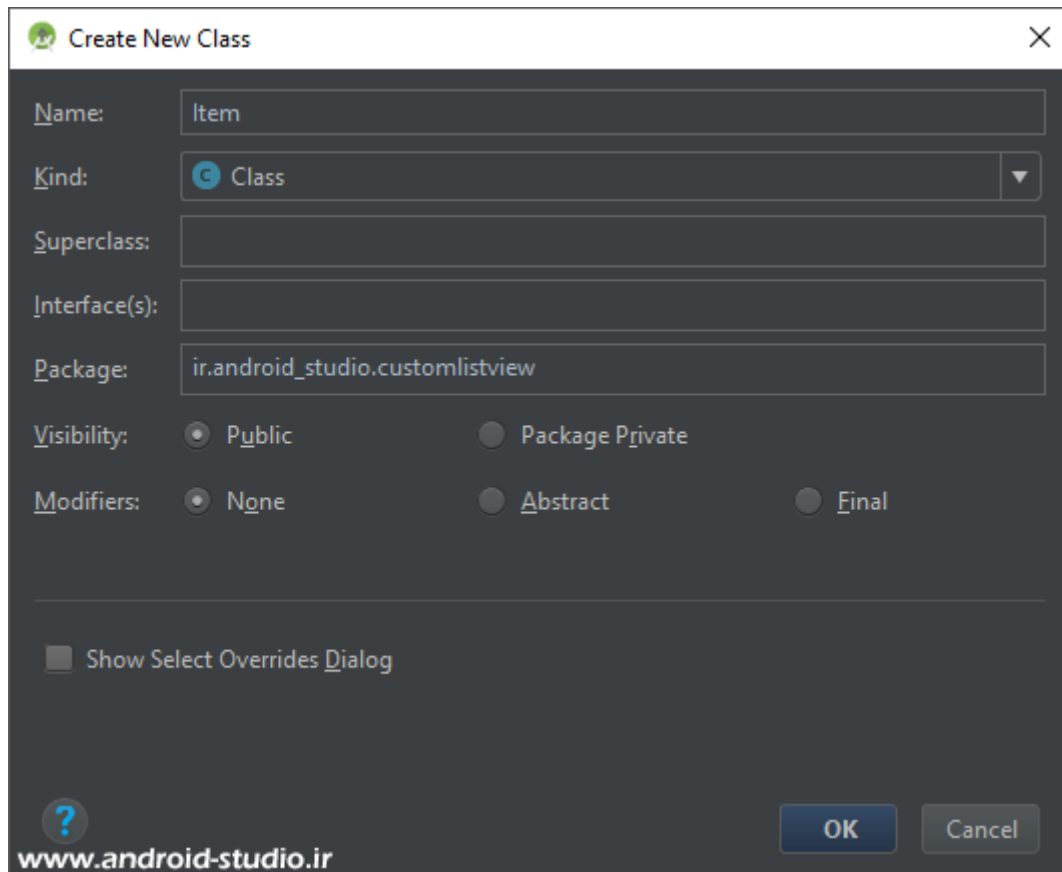
    <TextView
        android:id="@+id/txt_time"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Time" />

    <TextView
        android:id="@+id/txt_message"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Message" />

</LinearLayout>
```

برای مدیریت ویجت / view ها و ارتباط آنها با عملگرهای جاوا، از کلاسی استفاده می کنیم که اصطلاحاً Model نامیده می شود. یک کلاس با نام دلخواه Item.java به پکیج پروژه اضافه می کنیم:





سپس متغیرهای مدنظر را به کلاس اضافه می‌کنم:

```
package ir.android_studio.customlistview;

public class Item {

    private String imgAvatar;
    private String txtName;
    private String txtDate;
    private String txtMessage;

}
```

بعد از تعریف متغیرها، نوبت به اضافه کردن Constructor (متد سازنده) می‌رسد. علاوه بر نوشتن دستی متد سازنده، در اندروید استودیو امکان ساخت خودکار این متد نیز وجود دارد که با راست کلیک روی صفحه و انتخاب گزینه Generate یا کلیدهای ترکیبی alt + insert لیست زیر باز می‌شود:

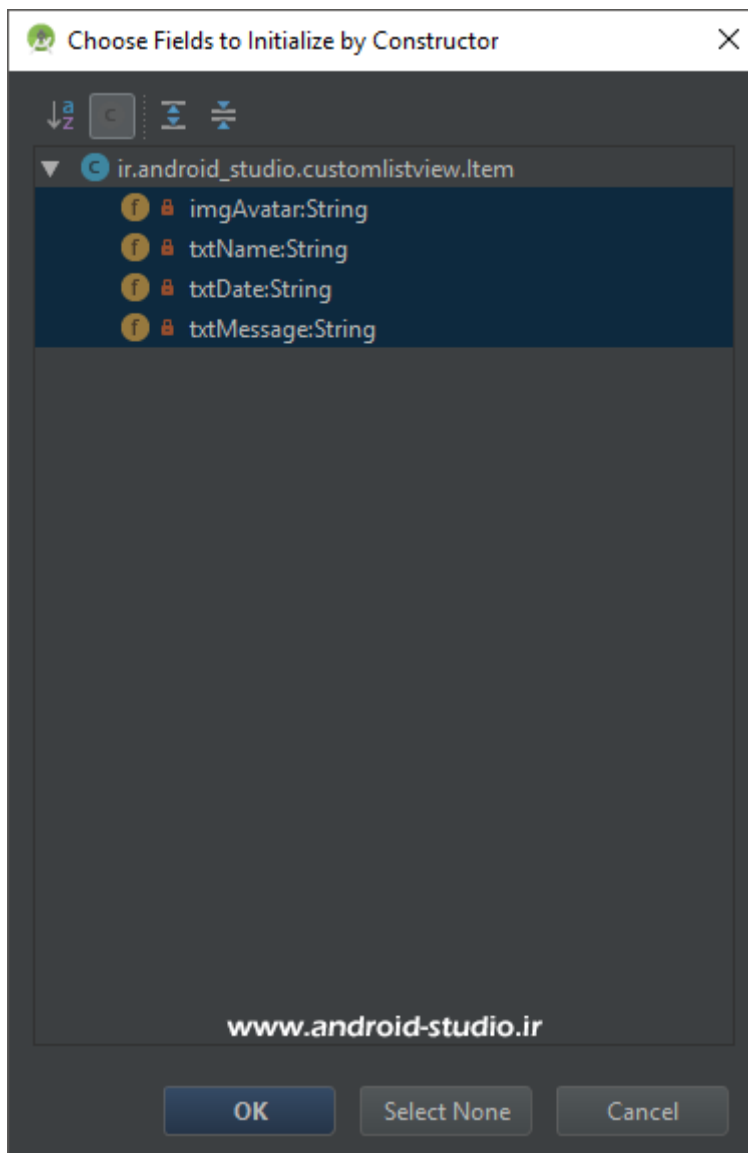


```
activity_main.xml x Item.java x MainActivity.java x
Item
1 package ir.android_studio.customlistview;
2
3 public class Item {
4
5     private String imgAvatar;
6     private String txtName;
7     private String txtDate;
8     private String txtMessage;
9
10
11 }
12
13
```

Generate

- Constructor
- Getter
- Setter
- Getter and Setter
- equals() and hashCode()
- toString()
- Override Methods... Ctrl+O
- Delegate Methods...
- Copyright

www.android-studio.ir



همه موارد را انتخاب کرده، تایید می کنم:

```
package ir.android_studio.customlistview;

public class Item {

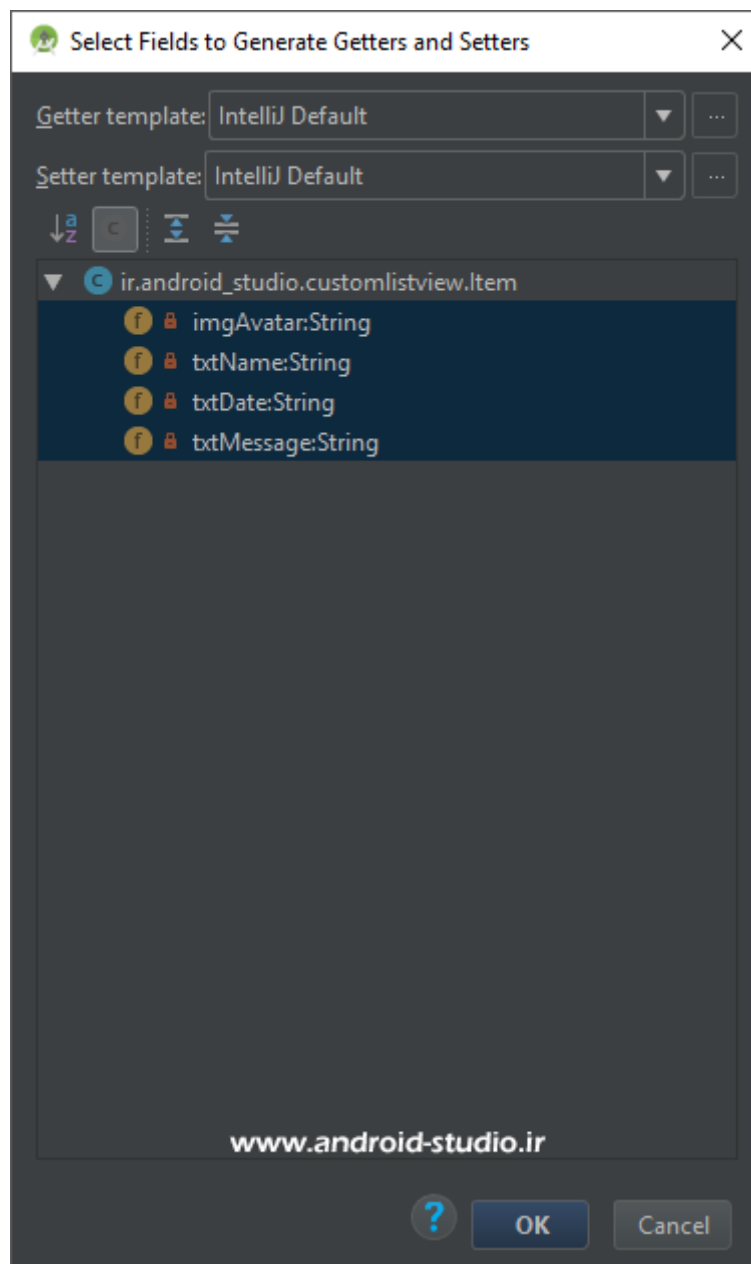
    private String imgAvatar;
    private String txtName;
    private String txtDate;
    private String txtMessage;

    public Item(String imgAvatar, String txtName, String txtDate, String txtMessage) {
        this.imgAvatar = imgAvatar;
        this.txtName = txtName;
        this.txtDate = txtDate;
        this.txtMessage = txtMessage;
    }
}
```



**نکته:** متد سازنده همیشه همانم با کلاس اصلی است و پارامترهای ورودی آن، متغیرهایی است که قبلا تعریف کرده ایم.

بعد از تعریف متد سازنده، متدهایی را باید به کلاس اضافه کنیم که به Getter and Setter مشهور هستند. مانند متد سازنده، این متدها را نیز می توان هم به صورت دستی و هم خودکار به کلاس اضافه کرد که این بار در پنجره Generate گزینه Getter and Setter را انتخاب و در مرحله بعد همه موارد را select میکنم:







:Item.java

```
package ir.android_studio.customlistview;

public class Item {

    private String imgAvatar;
    private String txtName;
    private String txtDate;
    private String txtMessage;

    public Item(String imgAvatar, String txtName, String txtDate, String txtMessage) {
        this.imgAvatar = imgAvatar;
        this.txtName = txtName;
        this.txtDate = txtDate;
        this.txtMessage = txtMessage;
    }

    public String getImgAvatar() {
        return imgAvatar;
    }

    public void setImgAvatar(String imgAvatar) {
        this.imgAvatar = imgAvatar;
    }

    public String getTxtName() {
        return txtName;
    }

    public void setTxtName(String txtName) {
        this.txtName = txtName;
    }

    public String getTxtDate() {
        return txtDate;
    }

    public void setTxtDate(String txtDate) {
        this.txtDate = txtDate;
    }

    public String getTxtMessage() {
        return txtMessage;
    }

    public void setTxtMessage(String txtMessage) {
        this.txtMessage = txtMessage;
    }
}
```

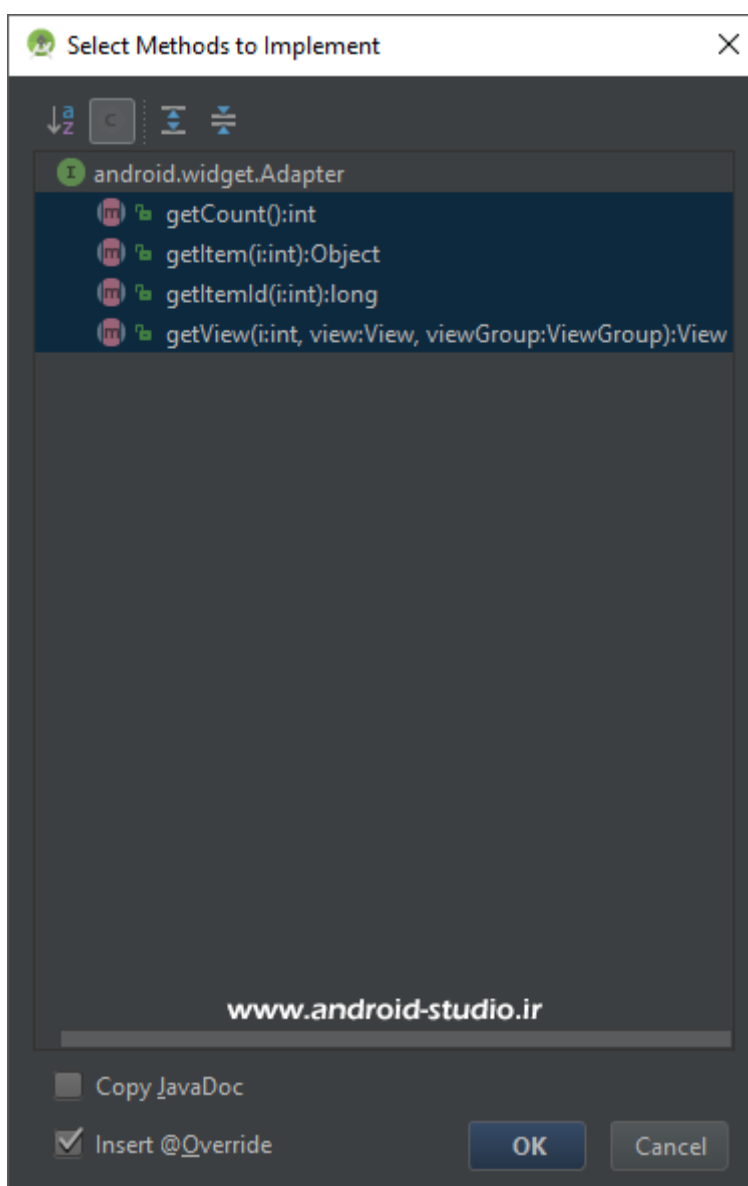
ملاحظه می کنید برای هر متغیر یک `get` و یک `set` ساخته شده که در مجموع برای چهار متغیر هشت متد داریم. نامگذاری متدهای `getter` و `setter` با فرمت `getVariableName` و `setVariableName` انجام می شود.

در پروژه قبل به دلیل اینکه فقط یک آرایه داشتیم، به سادگی و با استفاده از کلاس `ArrayAdapter`، درون اکتیویتی یک `Adapter` ایجاد کردیم. اما در این پروژه تعداد داده ها چهار نوع است بنابراین برای



Adapter هم یک کلاس مجزا می سازم. مانند قبل یک کلاس با نام دلخواه CustomAdapter.java به پکیج پروژه اضافه می کنم. این کلاس باید از یک Adapter ارث بری شود. در اندروید چند آداپتر داریم از جمله ArrayAdapter و BaseAdapter که من دومی را انتخاب می کنم. در ضمن ArrayAdapter نیز خودش از BaseAdapter ارث بری شده است.

بعد از extends کردن کلاس CustomAdapter از BaseAdapter اندروید استودیو اختاری مبنی بر Implement کردن متدهای موردنیاز BaseAdapter اعلام می کند که با کلیک روی علامت اختار و یا روی بدنه کلاس و کلید ترکیبی alt + Enter، انتخاب گزینه Implement Methods و در نهایت انتخاب هر چهار متد موجود در لیست، متدها به کلاس اضافه می شوند:





```
package ir.android_studio.customlistview;

import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;

public class CustomAdapter extends BaseAdapter {

    @Override
    public int getCount() {
        return 0;
    }

    @Override
    public Object getItem(int i) {
        return null;
    }

    @Override
    public long getItemId(int i) {
        return 0;
    }

    @Override
    public View getView(int i, View view, ViewGroup viewGroup) {
        return null;
    }
}
```

در مرحله بعد مانند کلاس Item.java متغیرهای موردنیاز و Constructor مربوطه را به کلاس اضافه می‌کنم:

```
private Context mContext;
private ArrayList<Item> mItem;

public CustomAdapter(Context mContext, ArrayList<Item> mItem) {
    this.mContext = mContext;
    this.mItem = mItem;
}
```

یک متغیر از جنس Context و نام دلخواه mContext و دیگری از جنس ArrayList و نوع Item (کلاسی که خودمان قبلاً ساختیم) با نام دلخواه mItem. و سپس نوشتن دستی متد سازنده یا ساخت خودکار آن توسط گزینه Generate در پنجره Constructor.



تا اینجا کار کلاس CustomAdapter به اینصورت تکمیل شد:

```
package ir.android_studio.customlistview;

import android.content.Context;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;

import java.util.ArrayList;

public class CustomAdapter extends BaseAdapter {

    private Context mContext;
    private ArrayList<Item> mItem;

    public CustomAdapter(Context mContext, ArrayList<Item> mItem) {
        this.mContext = mContext;
        this.mItem = mItem;
    }

    @Override
    public int getCount() {
        return 0;
    }

    @Override
    public Object getItem(int i) {
        return null;
    }

    @Override
    public long getItemId(int i) {
        return 0;
    }

    @Override
    public View getView(int i, View view, ViewGroup viewGroup) {
        return null;
    }

}
```

سه متد نخست را به اینصورت اصلاح می‌کنم:

```
@Override
public int getCount() {
    return mItem.size();
}

@Override
public Object getItem(int i) {
    return mItem.get(i);
}

@Override
public long getItemId(int i) {
    return i;
}
```



متد `getCount` از نوع `int` است و باید تعداد آیتم هایی که قرار است در `ListView` نمایش داده شود را برگرداند بنابراین می گویم `mItem.size()` را `return` کند. `getItem` هم مکان و یا شماره آیتم فعلی را بر می گرداند (یعنی آیتمی که نوبتش رسیده تا به `ListView` فرستاده شود).  
متد آخر یعنی `getView()` متد اصلی ماست که پردازش داده ها در این متد صورت می پذیرد. ابتدا کد کامل شده متد را به شما نشان داده و در ادامه به ارائه جزئیات می پردازم:

```
@Override
public View getView(int i, View view, ViewGroup viewGroup) {

    if (view == null) {
        view = LayoutInflater.from(mContext).inflate(R.layout.list_item, viewGroup, false);
    }

    Item currentItem = (Item) getItem(i);

    ImageView imgItemAvatar = view.findViewById(R.id.img_avatar);
    TextView txtItemName = view.findViewById(R.id.txt_name);
    TextView txtItemTime = view.findViewById(R.id.txt_time);
    TextView txtItemMessage = view.findViewById(R.id.txt_message);

    txtItemName.setText(currentItem.getTxtName());
    txtItemTime.setText(currentItem.getTxtDate());
    txtItemMessage.setText(currentItem.getTxtMessage());

    String mUri = "@drawable/" + currentItem.getImgAvatar();
    int imageSource = mContext.getResources().getIdentifier(mUri, null,
mContext.getPackageName());
    imgItemAvatar.setImageDrawable(ResourcesCompat.getDrawable(mContext.getResources(),
imageSource, null));

    return view;
}
```

در قسمت اول یک `if` داریم که گفته تا زمانی که `view` برابر `null` باشد، دستور داخل حلقه را اجرا کن. یعنی تا زمانی که صفحه نمایش جا برای اضافه شدن آیتم جدید به `ListView` دارد، توسط `LayoutInflater` یک داده جدید به لایه `list_item` که حاوی یک `ImageView` و سه `TextView` هست بفرست.

در خط بعد از کلاس `Item` یک نمونه جدید با نام دلخواه `currentItem` ساختم که توسط `getItem(i)` موقعیت داده (آیتم) فعلی که در حال اضافه شدن به `ListView` هست را مشخص می کند. در ادامه ویجت های نمایش داده ها را تعریف می کنم.

در قسمت بعد هم انتقال داده به `view` انجام می شود.



```
txtItemName.setText(currentItem.getTxtName());
```

در خط بالا ابتدا تکست مربوط به Name آئیم در حال پردازش گرفته می شود (currentItem.getTxtName)، سپس توسط setText به txtItemName ارسال می گردد. برای دو مورد دیگر یعنی تاریخ (txtDate) و متن پیام (txtMessage) به همین ترتیب.

سه خط بعد مربوط به ساخت و ست کردن آدرس عکس است. در خط اول، مسیر تصویر که @drawable به علاوه نام تصویر آئیم فعلی است در mUri ریخته می شود. دو خط بعد مقداری از توضیحات این مبحث خارج است و لزومی به حفظ و یادگیری آن نیست. فقط بدانید برای دریافت و ارسال آدرس تصاویر به این شیوه عمل می کنیم.

و در نهایت return view.

فعلا با کلاس Adapter کاری ندارم. به سراغ اکتیویتی می روم.

```
package ir.android_studio.customlistview;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ListView;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

    ListView mList;
    ArrayList<Item> arrayItem;

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

        mList = findViewById(R.id.list_view);

        arrayItem = new ArrayList<>();

        CustomAdapter mAdapter = new CustomAdapter(this, arrayItem);

        mList.setAdapter(mAdapter);
    }
}
```

در خط اول ویجت ListView تعریف شده. سپس یک نمونه از ArrayList با نام دلخواه arrayItem ایجاد کرده ام. در خط بعد یک نمونه از کلاس CustomAdapter ساخته شده که ورودی اول this



برای Context و ورودی دوم همان نمونه ای است که از ArrayList ساختیم. در نهایت mAdapter توسط متد setAdapter به ListView متصل می شود.

حالا تنها کاری که مانده، تعریف چند آیتم برای ListView است:

```

mList = findViewById(R.id.list_view);

arrayItem = new ArrayList<>();

CustomAdapter mAdapter = new CustomAdapter( mContext: this, arrayItem);

arrayItem.add(new Item());

```

String imgAvatar, String txtName, String txtDate, String txtMessage

www.android-studio.ir

تعریف آیتم ها توسط add انجام می شود. داخل پرانتز بعد از new Item() مشاهده می کنید همان پارامترهای ورودی متد سازنده Item.java را از ما می خواهد.

چند تصویر به فولدر drawable اضافه می کنم. سپس به تعداد تصاویر، آیتم آزمایشی تعریف می کنم:

```

arrayItem.add(new Item("avatar_1", "SeyedMahdi", "12:14", "سلام. بزم آبگرم فردوس؟"));
arrayItem.add(new Item("avatar_2", "Ahmad", "11:41", "فایلا رو دانلود کردی؟"));
arrayItem.add(new Item("avatar_3", "Morteza", "11:38", "شرمنده امروز نمیرسم پیام"));
arrayItem.add(new Item("avatar_4", "Farhad", "11:32", "اوکی. ممنون"));

```

اگر بخاطر داشته باشید هنگام ساخت mUri رشته @drawable را به ابتدای مسیر اضافه کردیم بنابراین در اینجا فقط کافیسست نام تصویر را وارد کنیم.



## :MainActivity.java

```
package ir.android_studio.customlistview;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ListView;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

    ListView mList;
    ArrayList<Item> arrayItem;

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

        mList = findViewById(R.id.list_view);

        arrayItem = new ArrayList<>();

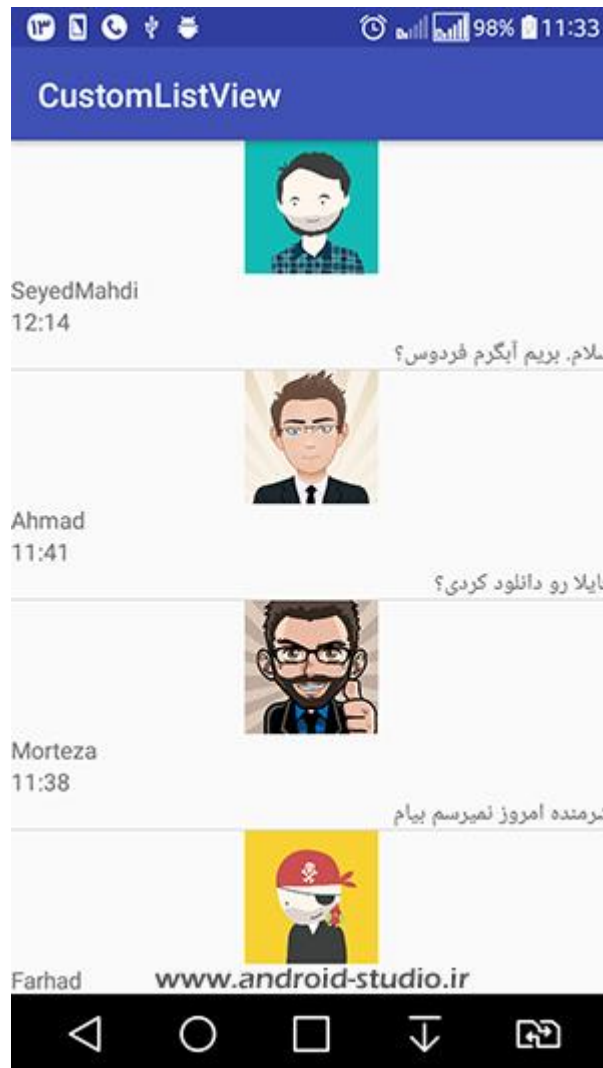
        CustomAdapter mAdapter = new CustomAdapter(this, arrayItem);

        arrayItem.add(new Item("avatar_1", "SeyedMahdi", "12:14", "فردوس؟ آبگرم بریم .سلام");
        arrayItem.add(new Item("avatar_2", "Ahmad", "11:41", "کردی؟ دانلود رو فایلا");
        arrayItem.add(new Item("avatar_3", "Morteza", "11:38", "پیام نمیرسم امروز شرمنده");
        arrayItem.add(new Item("avatar_4", "Farhad", "11:32", "ممنون .اوکی");

        mList.setAdapter(mAdapter);
    }
}
```

پروژه را اجرا می کنیم:





لیست با موفقیت ساخته شد و ۴ آیتم به ترتیب در زیر هم قرار گرفته اند. به دلیل اینکه ویجت های list\_item را به صورت ساده تعریف کردم، آیتم ها ظاهر بهم ریخته ای دارند که با استفاده از لایه های تو در تو LinearLayout و وزن دهی، چینش عناصر را انجام می دهیم:



```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="100dp"
    android:padding="5dp"
    android:orientation="horizontal">

    <ImageView
        android:id="@+id/img_avatar"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="0.2"
        android:src="@drawable/avatar_1" />

    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="0.8"
        android:orientation="vertical">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="0.5"
            android:orientation="horizontal">

            <TextView
                android:id="@+id/txt_name"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:gravity="bottom"
                android:paddingLeft="5dp"
                android:text="Name" />

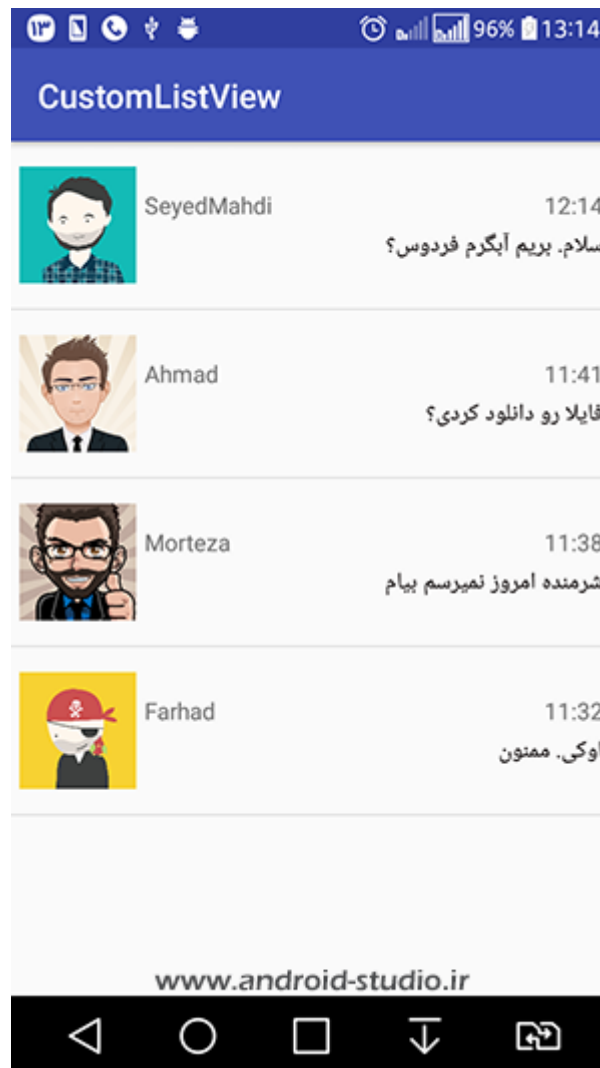
            <TextView
                android:id="@+id/txt_time"
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:gravity="bottom|right"
                android:text="Time" />
        </LinearLayout>

        <TextView
            android:id="@+id/txt_message"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_marginTop="5dp"
            android:layout_weight="0.5"
            android:paddingLeft="5dp"
            android:text="Message"
            android:textColor="#161111" />
    </LinearLayout>
</LinearLayout>
```

کار با LinearLayout و سایر لایه ها در [فصل پنجم](#) بررسی شده و در این مبحث به جزئیات طراحی لایه نمی پردازم.



مجدد پروژه را اجرا می کنم:



محیط ListView ظاهری شبیه به پیام رسان های اجتماعی به خود گرفته و به نتیجه مطلوب رسیدیم. برای مرتب کد اکتیویتی، داده ها را به داخل یک تابع مستقل انتقال داده و تابع را در جایگاه فعلی داده ها صدا می زنم:



```

package ir.android_studio.customlistview;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ListView;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

    ListView mList;
    ArrayList<Item> arrayItem;

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

        mList = findViewById(R.id.list_view);

        arrayItem = new ArrayList<>();

        CustomAdapter mAdapter = new CustomAdapter(this, arrayItem);

        itemDetails();

        mList.setAdapter(mAdapter);
    }

    public void itemDetails() {

        arrayItem.add(new Item("avatar_1", "SeyedMahdi", "12:14", "سلام .بریم .آبگرم ؟ فردوس؟"));
        arrayItem.add(new Item("avatar_2", "Ahmad", "11:41", "دانلود رو فایل؟ کردی؟"));
        arrayItem.add(new Item("avatar_3", "Morteza", "11:38", "بیم نمیرسم امروز شرمنده"));
        arrayItem.add(new Item("avatar_4", "Farhad", "11:32", "ممنون .اوکی"));
        arrayItem.add(new Item("avatar_1", "SeyedMahdi", "12:14", "سلام .بریم .آبگرم ؟ فردوس؟"));
        arrayItem.add(new Item("avatar_2", "Ahmad", "11:41", "دانلود رو فایل؟ کردی؟"));
        arrayItem.add(new Item("avatar_3", "Morteza", "11:38", "بیم نمیرسم امروز شرمنده"));
        arrayItem.add(new Item("avatar_4", "Farhad", "11:32", "ممنون .اوکی"));

    }
}

```

تعداد آیتم ها را یکبار تکرار کردم تا اسکرول شدن لیست هم تست شود.

برنامه ما به درستی کار می کند و مشکلی ندارد. اما می توانیم قدری کدها را بهینه تر کنیم تا حافظه کمتری از دیوایس کاربر هنگام ساخت آیتم های ListView مصرف شود.

تغییراتی در کلاس CustomAdapter به صورت زیر اعمال می کنم:



```
private class ViewHolder {

    ImageView imgItemAvatar;
    TextView txtItemName;
    TextView txtItemTime;
    TextView txtItemMessage;

    public ViewHolder(View mView) {
        imgItemAvatar = mView.findViewById(R.id.img_avatar);
        txtItemName = mView.findViewById(R.id.txt_name);
        txtItemTime = mView.findViewById(R.id.txt_time);
        txtItemMessage = mView.findViewById(R.id.txt_message);
    }
}

@Override
public View getView(int i, View view, ViewGroup viewGroup) {

    ViewHolder vHolder;

    if (view == null) {
        view = LayoutInflater.from(mContext).inflate(R.layout.list_item, viewGroup, false);
        vHolder = new ViewHolder(view);
        view.setTag(vHolder);
    } else {
        vHolder = (ViewHolder) view.getTag();
    }

    Item currentItem = (Item) getItem(i);

    vHolder.txtItemName.setText(currentItem.getTxtName());
    vHolder.txtItemTime.setText(currentItem.getTxtDate());
    vHolder.txtItemMessage.setText(currentItem.getTxtMessage());

    String mUri = "@drawable/" + currentItem.getImgAvatar();
    int imageSource = mContext.getResources().getIdentifier(mUri, null,
mContext.getPackageName());

    vHolder.imgItemAvatar.setImageDrawable(ResourcesCompat.getDrawable(mContext.getResources(),
imageSource, null));

    return view;
}
```

قبل از متد `getView` یک کلاس با نام `ViewHolder` اضافه کردم. ملاحظه می کنید اینبار ویجت ها در کلاس `ViewHolder` تعریف شده اند نه `getView`. در حلقه `if else` یک `setTag` و یک `getTag` داریم. تا زمانی که صفحه نمایش جا برای اضافه شدن آیتم جدید دارد، آیتم ها `setTag` می شوند و بعد از پر شدن صفحه، آیتم های بعد `getTag` شده و در اینجا نگه داری می شوند تا زمانی که کاربر به پایین یا بالا اسکرول کند تا این آیتم ها مجدد به صفحه نمایش برگردند. کد جدید از این جهت نسبت به کد قبل بهینه است که حالا فقط یکبار و درون کلاس `ViewHolder` ویجت ها تعریف می شوند و این عمل برای هر آیتم جداگانه تکرار نمی شود.



## کد کامل CustomAdapter.java:

```
package ir.android_studio.customlistview;

import android.content.Context;
import android.support.v4.content.res.ResourcesCompat;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.ImageView;
import android.widget.TextView;

import java.util.ArrayList;

public class CustomAdapter extends BaseAdapter {

    private Context mContext;
    private ArrayList<Item> mItem;

    public CustomAdapter(Context mContext, ArrayList<Item> mItem) {
        this.mContext = mContext;
        this.mItem = mItem;
    }

    @Override
    public int getCount() {
        return mItem.size();
    }

    @Override
    public Object getItem(int i) {
        return mItem.get(i);
    }

    @Override
    public long getItemId(int i) {
        return i;
    }

    private class ViewHolder {

        ImageView imgItemAvatar;
        TextView txtItemName;
        TextView txtItemTime;
        TextView txtItemMessage;

        public ViewHolder(View mView) {
            imgItemAvatar = mView.findViewById(R.id.img_avatar);
            txtItemName = mView.findViewById(R.id.txt_name);
            txtItemTime = mView.findViewById(R.id.txt_time);
            txtItemMessage = mView.findViewById(R.id.txt_message);
        }

    }

    @Override
    public View getView(int i, View view, ViewGroup viewGroup) {

        ViewHolder vHolder;

        if (view == null) {
            view = LayoutInflater.from(mContext).inflate(R.layout.list_item, viewGroup,
false);
            vHolder = new ViewHolder(view);
            view.setTag(vHolder);
        }
    }
}
```



```

    } else {
        viewHolder = (ViewHolder) view.getTag();
    }

    Item currentItem = (Item) getItem(i);

    viewHolder.txtItemName.setText(currentItem.getTxtName());
    viewHolder.txtItemTime.setText(currentItem.getTxtDate());
    viewHolder.txtItemMessage.setText(currentItem.getTxtMessage());

    String mUri = "@drawable/" + currentItem.getImgAvatar();
    int imageSource = mContext.getResources().getIdentifier(mUri, null,
mContext.getPackageName());

    viewHolder.imgItemAvatar.setImageDrawable(ResourcesCompat.getDrawable(mContext.getResources(),
imageSource, null));

    return view;
}
}

```

**توجه:** به جهت دسترسی ساده به هر دو کد، سورس قبلی کلاس CustomAdapter با نام CustomAdapter\_backup.java در کنار سورس پروژه قرار داده شده است.

آخرین موردی که بررسی می کنیم، کلیک روی آیتم هاست.

در MainActivity.java و درون متد onCreate یک setOnItemClickListener ایجاد می کنیم:

```

itemDetails();
mList.setAdapter(@Nullable OnItemClickListener listener);
mList.setOnItemClickListener(new Adapter);
}

```

www.android-studio.ir

AdapterView.OnItemClickListener(...) (android.widget.AdapterVi...  
CustomAdapter (ir.android\_studio.customlistview)

```

mList.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {
    }
});

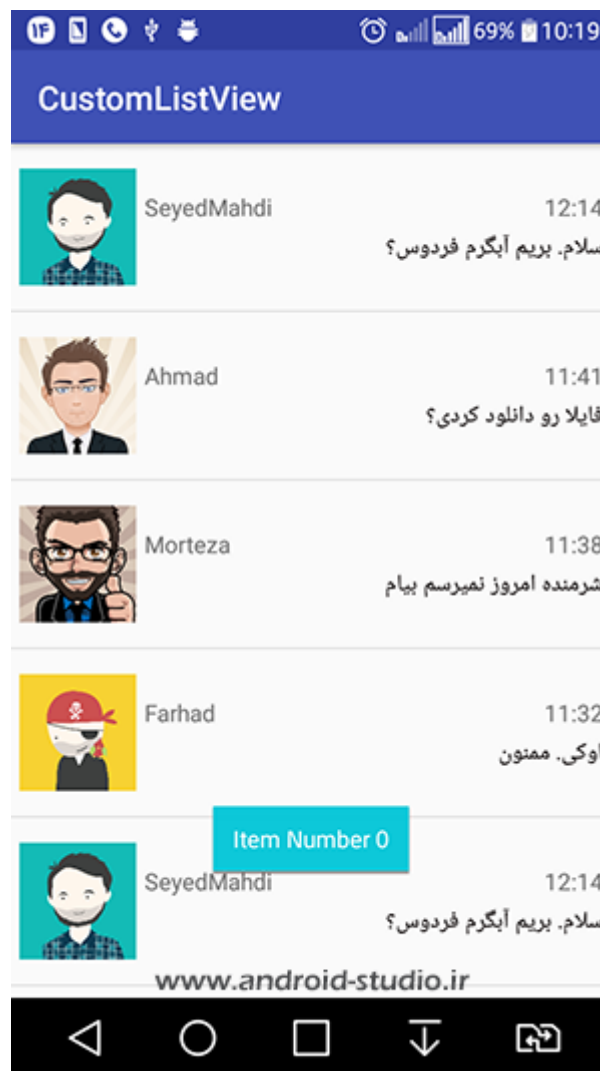
```

به جهت تست این متد یک Toast ساده به آن اضافه می کنیم:



```
mList.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {
        Toast.makeText(MainActivity.this, "Item Number " + i, Toast.LENGTH_SHORT).show();
    }
});
```

پروژه را اجرا می‌کنم. با لمس هر آیتم، Toast مربوط با آن نمایش داده می‌شود که همان شماره و شناسه آیتم است (از صفر شروع می‌شود):



**توجه:** سورس پروژه ها درون پوشه Exercises قرار داده شده است

با ارائه انتقادات و پیشنهادات خود، ما را در ارائه آموزش های بهتر یاری فرمائید.

[www.android-studio.ir](http://www.android-studio.ir)