[Список-дерево ExpandableListView](https://startandroid.ru/ru/uroki/vse-uroki-spiskom/86-urok-45-spisok-derevo-expandablelistview.html" \o "Урок 45. Список-дерево ExpandableListView)

- строим список-дерево ExpandableListView

Если список элементов получается большой, имеет смысл разбить его на **группы** для упрощения навигации. Для этих целей можно использовать [ExpandableListView](http://developer.android.com/reference/android/widget/ExpandableListView.html" \t "_blank). Это список в виде двухуровневого дерева. Первый уровень – **группа**, а в ней второй – **элемент**.

Чтобы построить такой список нам нужно как-то передать адаптеру данные по группам и элементам.

Каждая группа представляет из себя **Map<String, ?>**. Этот Map содержит **атрибуты**, которые вам нужны для каждой группы. Потом все эти Map (группы) собираются в List-**коллекцию**, например ArrayList. В итоге мы получили упакованные в один объект группы.

Каждый элемент группы также представлен объектом **Map<String, ?>**. Мы собираем все Map (элементы) для каждой группы в отдельную **коллекцию**. Получается, каждой группе соответствует коллекция с элементами. Далее эти коллекции мы теперь помещаем в общую коллекцию. Т.е. получается подобие двумерного массива. И в итоге пункты упакованы в один объект.

Сейчас начнем кодить пример и там станет понятнее.

*List-коллекции называются обычно «список». Но т.к. список в контексте последних уроков - это набор пунктов на экране (ListView), то чтобы не путаться я буду использовать слово «коллекция».*

Создадим проект:

**Project name**: P0451\_ExpandableList  
**Build Target**: Android 2.3.3  
**Application name**: ExpandableList  
**Package name**: ru.startandroid.develop.p0451expandablelist  
**Create Activity**: MainActivity

Нарисуем экран **main.xml**:

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <LinearLayout      xmlns:android="http://schemas.android.com/apk/res/android"      android:layout\_width="fill\_parent"      android:layout\_height="fill\_parent"      android:orientation="vertical">      <ExpandableListView          android:id="@+id/elvMain"          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content">      </ExpandableListView>  </LinearLayout> |

 Только **ExpandableList**на экране.

Код **MainActivity.java**:

|  |
| --- |
| package ru.startandroid.develop.p0451expandablelist;    import java.util.ArrayList;  import java.util.HashMap;  import java.util.Map;    import android.app.Activity;  import android.os.Bundle;  import android.widget.ExpandableListView;  import android.widget.SimpleExpandableListAdapter;    public class MainActivity extends Activity {      // названия компаний (групп)    String[] groups = new String[] {"HTC", "Samsung", "LG"};      // названия телефонов (элементов)    String[] phonesHTC = new String[] {"Sensation", "Desire", "Wildfire", "Hero"};    String[] phonesSams = new String[] {"Galaxy S II", "Galaxy Nexus", "Wave"};    String[] phonesLG = new String[] {"Optimus", "Optimus Link", "Optimus Black", "Optimus One"};      // коллекция для групп    ArrayList<Map<String, String>> groupData;      // коллекция для элементов одной группы    ArrayList<Map<String, String>> childDataItem;      // общая коллекция для коллекций элементов    ArrayList<ArrayList<Map<String, String>>> childData;    // в итоге получится childData = ArrayList<childDataItem>      // список атрибутов группы или элемента    Map<String, String> m;      ExpandableListView elvMain;          /\*\* Called when the activity is first created. \*/      public void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.main);            // заполняем коллекцию групп из массива с названиями групп          groupData = new ArrayList<Map<String, String>>();          for (String group : groups) {            // заполняем список атрибутов для каждой группы            m = new HashMap<String, String>();              m.put("groupName", group); // имя компании              groupData.add(m);          }            // список атрибутов групп для чтения          String groupFrom[] = new String[] {"groupName"};          // список ID view-элементов, в которые будет помещены атрибуты групп          int groupTo[] = new int[] {android.R.id.text1};              // создаем коллекцию для коллекций элементов          childData = new ArrayList<ArrayList<Map<String, String>>>();            // создаем коллекцию элементов для первой группы          childDataItem = new ArrayList<Map<String, String>>();          // заполняем список атрибутов для каждого элемента          for (String phone : phonesHTC) {            m = new HashMap<String, String>();              m.put("phoneName", phone); // название телефона              childDataItem.add(m);          }          // добавляем в коллекцию коллекций          childData.add(childDataItem);            // создаем коллекцию элементов для второй группы          childDataItem = new ArrayList<Map<String, String>>();          for (String phone : phonesSams) {            m = new HashMap<String, String>();              m.put("phoneName", phone);              childDataItem.add(m);          }          childData.add(childDataItem);            // создаем коллекцию элементов для третьей группы          childDataItem = new ArrayList<Map<String, String>>();          for (String phone : phonesLG) {            m = new HashMap<String, String>();              m.put("phoneName", phone);              childDataItem.add(m);          }          childData.add(childDataItem);            // список атрибутов элементов для чтения          String childFrom[] = new String[] {"phoneName"};          // список ID view-элементов, в которые будет помещены атрибуты элементов          int childTo[] = new int[] {android.R.id.text1};            SimpleExpandableListAdapter adapter = new SimpleExpandableListAdapter(              this,              groupData,              android.R.layout.simple\_expandable\_list\_item\_1,              groupFrom,              groupTo,              childData,              android.R.layout.simple\_list\_item\_1,              childFrom,              childTo);            elvMain = (ExpandableListView) findViewById(R.id.elvMain);          elvMain.setAdapter(adapter);      }  } |

Код громоздкий и сложный, давайте разбираться.

Сначала мы в классе описываем массивы **данных** – это названия **групп** и названия **элементов** для них. Я решил в качестве данных выбрать смартфоны. **Группы**в нашем списке – это будут **компании**, а **элементы**– **смартфоны** этих компаний.

Затем описываем коллекцию для групп, коллекции для элементов и Map для атрибутов.

В методе onCreate заполняем **groupData**. Это коллекция групп. Каждая группа представляет собой **Map**. А в Map мы пишем необходимые нам **атрибуты**для каждой группы. В нашем случае, для каждой группы мы укажем всего один атрибут **groupName** - это **название** компании из массива **groups**.

Как мы помним, **адаптер**обычно использует **layout**-ресурс для отображения пункта списка. В нашем случае **пунктами**ListView являются и **группа**и **элемент**.  В layout-ресурсе могут быть какие-либо **TextView**. Мы можем заполнить их значениями из **атрибутов** элементов или групп, которые собраны в Map. Для этого нам надо указать сначала **имена** **атрибутов**, которые хотим использовать, а затем **ID TextView**-элементов, в которые хотим поместить значения этих атрибутов. Речь сейчас идет о текстовых атрибутах. (Хотя вообще атрибут вовсе не обязан быть класса String)

Для связки **атрибутов**и **TextView**-элементов мы используем два массива:

**groupFrom**– список имен атрибутов, которые будут считаны. В нашем случае – это **groupName**, который мы добавили к группе с помощью Map чуть выше в коде, когда собирали группы в groupData.  
**groupTo**– список ID View-элементов, в которые будут помещены считанные значения атрибутов. Наш используемый layout будет содержать TextView с ID = **android.R.id.text1**.

Два этих массива сопоставляются по порядку элементов. В итоге, в **layout**-ресурсе группы найдется элемент с ID = **android.R.id.text1** и в него запишется текст из атрибута **groupName**. Тем самым мы получим отображение имени группы (компании) в списке.

Далее формируем **коллекции элементов**. Создаем общую **коллекцию коллекций**. А затем создаем коллекции элементов каждой группы. Принцип тот же, что и с группами – создаем **Map**и в него пишем атрибут **phoneName**со значением равным **имени**элемента (телефона). Коллекцию элементов для каждой группы добавляем в общую коллекцию.

Формируем два массива для сопоставления **TextView**из layout и **атрибутов**элементов. Полностью аналогично, как выше мы уже проделали с группами. В итоге при отображении элемента, найдется TextView с ID = android.R.id.text1 и туда запишется текст из атрибута phoneName. И мы увидим текст нашего элемента (телефона) в списке.

В конце кода мы создаем адаптер [SimpleExpandableListAdapter](http://developer.android.com/reference/android/widget/SimpleExpandableListAdapter.html" \t "_blank) и присваиваем его списку.

На вход при создании адаптера идут элементы:

**this**– контекст   
**groupData**– коллекция групп  
**android.R.layout.simple\_expandable\_list\_item\_1** – layout-ресурс, который будет использован для отображения группы в списке. Соответственно, запросто можно использовать свой layout-файл.  
**groupFrom**– массив имен атрибутов групп  
**groupTo**– массив ID TextView из layout для групп  
**childData**– коллекция коллекций элементов по группам  
**android.R.layout.simple\_list\_item\_1** - layout-ресурс, который будет использован для отображения элемента в списке. Можно использовать свой layout-файл  
**childFrom**– массив имен атрибутов элементов  
**childTo**- массив ID TextView из layout для элементов.

В общем непростая, на мой взгляд, реализация дерева получилась. Возможно, не сразу получится понять.  Но я попытался расписать все досконально и как можно подробнее.

Layout **simple\_expandable\_list\_item\_1**, который мы использовали для отображения групп – это TextView с отступом от левого края, чтобы осталось место для кнопки раскрытия/сворачивания списка.  Для эксперимента вы можете попробовать использовать для групп layout **simple\_list\_item\_1**, который мы использовали для элементов. В этом случае текст будет пересекаться с кнопкой.

А вообще вы можете создать для элементов свой **layout**, например, с тремя **TextView**. И к каждому элементу списка (Map) добавить еще по два атрибута: цена и цвет. Далее указываете ваш layout в конструкторе, формируете соответственно массивы **childFrom**и **childTo** чтобы сопоставить атрибуты и TextView, и получится, что каждый элемент группы содержит более подробную информацию о смартфоне.