Списки и адаптеры

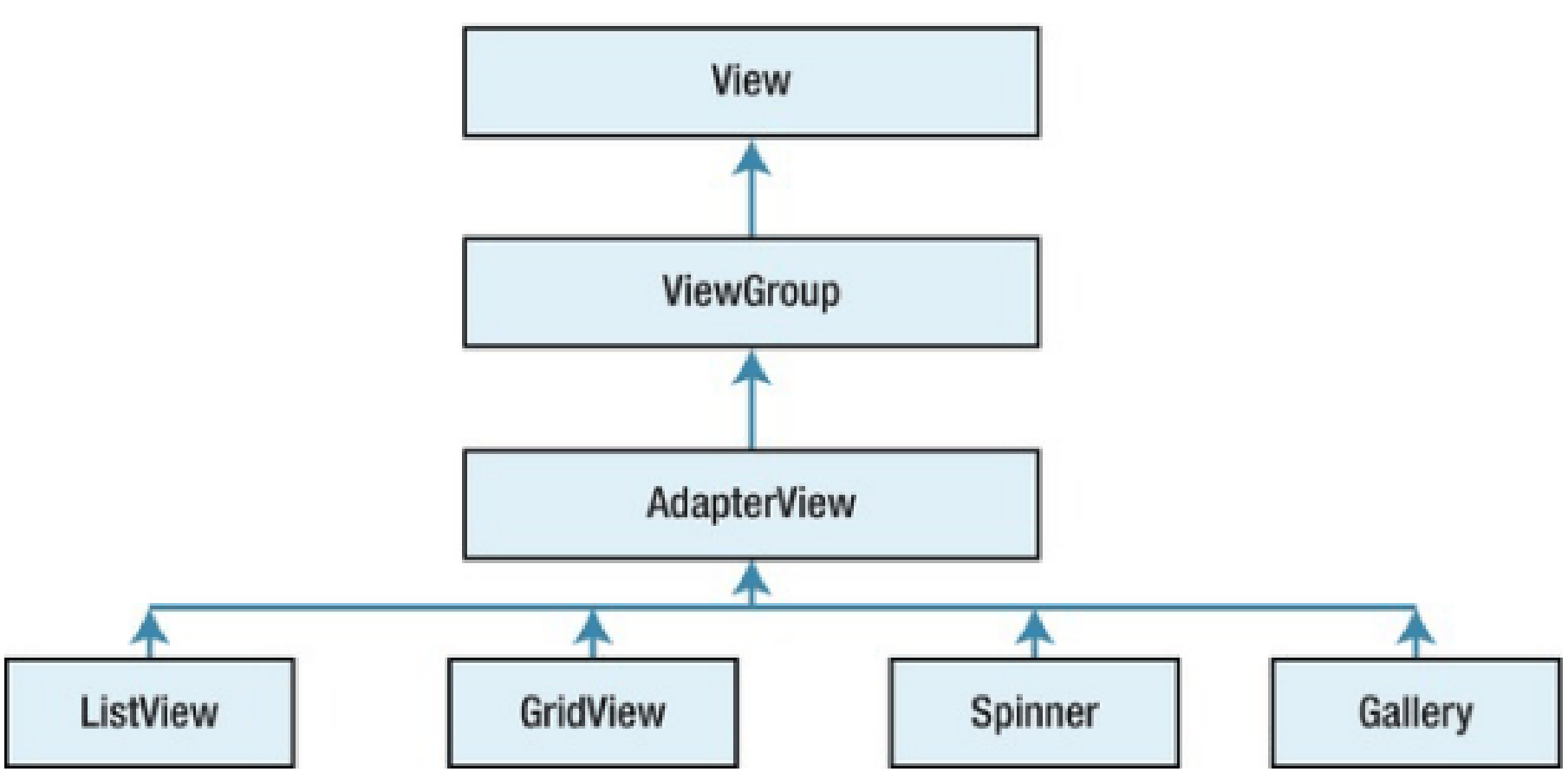
Списки

Android представляет широкую палитру различных списков. Все они является наследниками класса **android.widget.AdapterView**. Это такие виджеты:

- **ListView**
- **GridView**
- **Spinner**

Они могут выступать контейнерами для других элементов управления.

Адаптеры и списки



3 компонента

При работе со списками мы имеем дело с тремя компонентами.

1. Элемент списка (ListView, GridView, …)

2. Источник данных (массив, объект ArrayList, база данных и т.д.)

3. Адаптеры - специальные компоненты, которые связывают источник данных с элементом списка

ArrayAdapter

Класс **ArrayAdapter** представляет собой простейший адаптер, который связывает массив данных с набором элементов TextView, из которых, к примеру, может состоять ListView. То есть в данном случае источником данных выступает массив объектов. ArrayAdapter вызывает у каждого объекта метод toString() для приведения к строковому виду и полученную строку устанавливает в элемент TextView.

ArrayAdapter. Пример

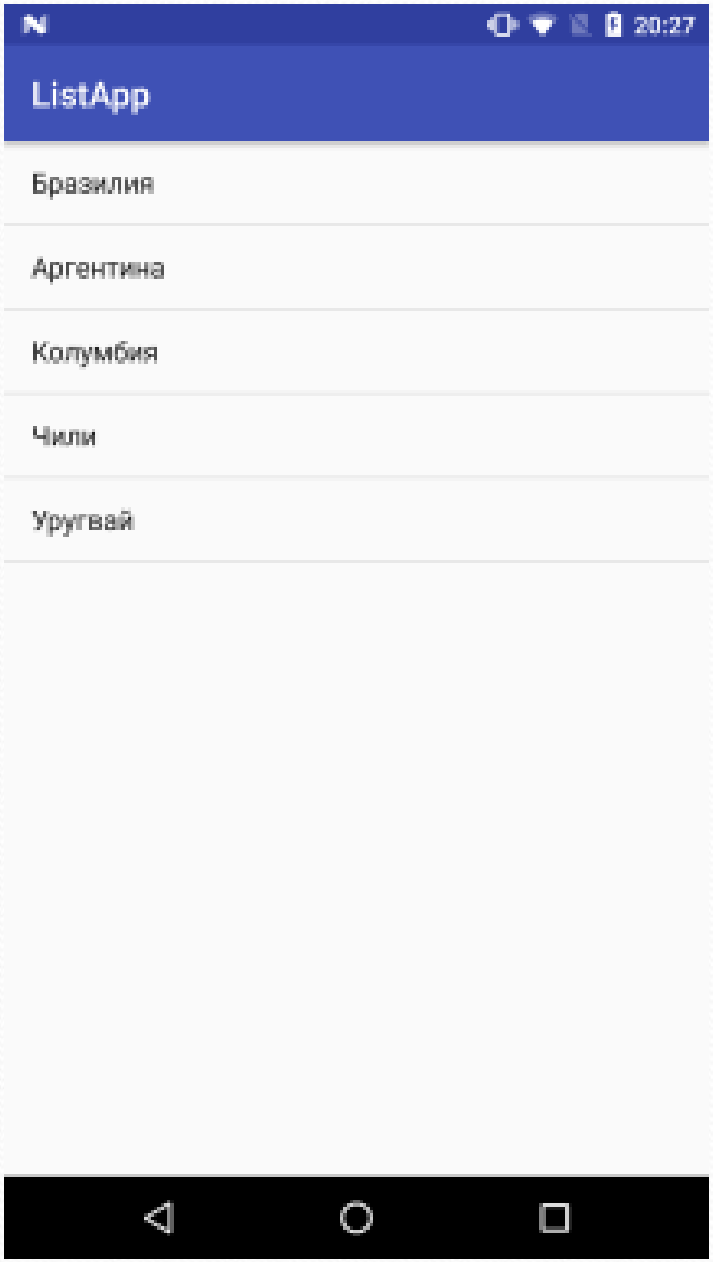
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

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

</RelativeLayout>
```

```
8  public class MainActivity extends AppCompatActivity {
9
10     // набор данных, которые свяжем со списком
11     String[] countries = { "Бразилия", "Аргентина", "Колумбия", "Чили", "Уругвай"};
12     @Override
13     protected void onCreate(Bundle savedInstanceState) {
14         super.onCreate(savedInstanceState);
15         setContentView(R.layout.activity_main);
16
17         // получаем элемент ListView
18         ListView countriesList = (ListView) findViewById(R.id.countriesList);
19
20         // создаем адаптер
21         ArrayAdapter<String> adapter = new ArrayAdapter(this,
22                 android.R.layout.simple_list_item_1, countries);
23
24         // устанавливаем для списка адаптер
25         countriesList.setAdapter(adapter);
26     }
27 }
```

ArrayAdapter

Для создания адаптера использовался следующий конструктор ArrayAdapter<String>(this,android.R.layout.simple_list_item_1, countries), где

- **this** : текущий объект activity
- **android.R.layout.simple_list_item_1** : файл разметки списка, который фреймворк представляет по умолчанию. Он находится в папке Android SDK по пути platforms/[android-номер_версии]/data/res/layout. Если нас не удовлетворяет стандартная разметка списка, мы можем создать свою и потом в коде изменить id на id нужной нам разметки
- **countries** : массив данных. Здесь необязательно указывать именно массив, это может быть список ArrayList<T>.

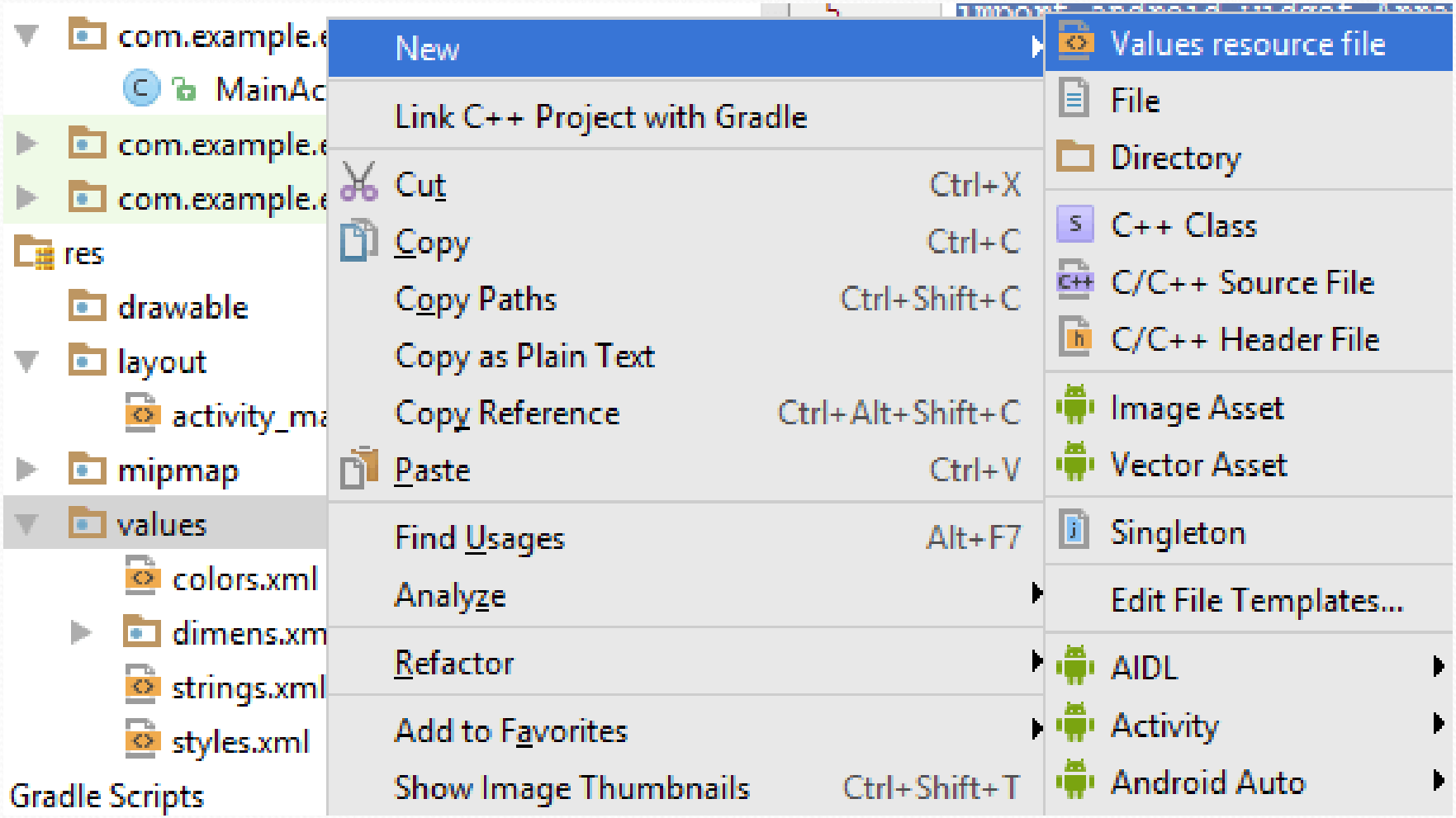

20:27
ListApp
Бразилия
Аргентина
Колумбия
Чили
Уругвай

Ресурс string-array

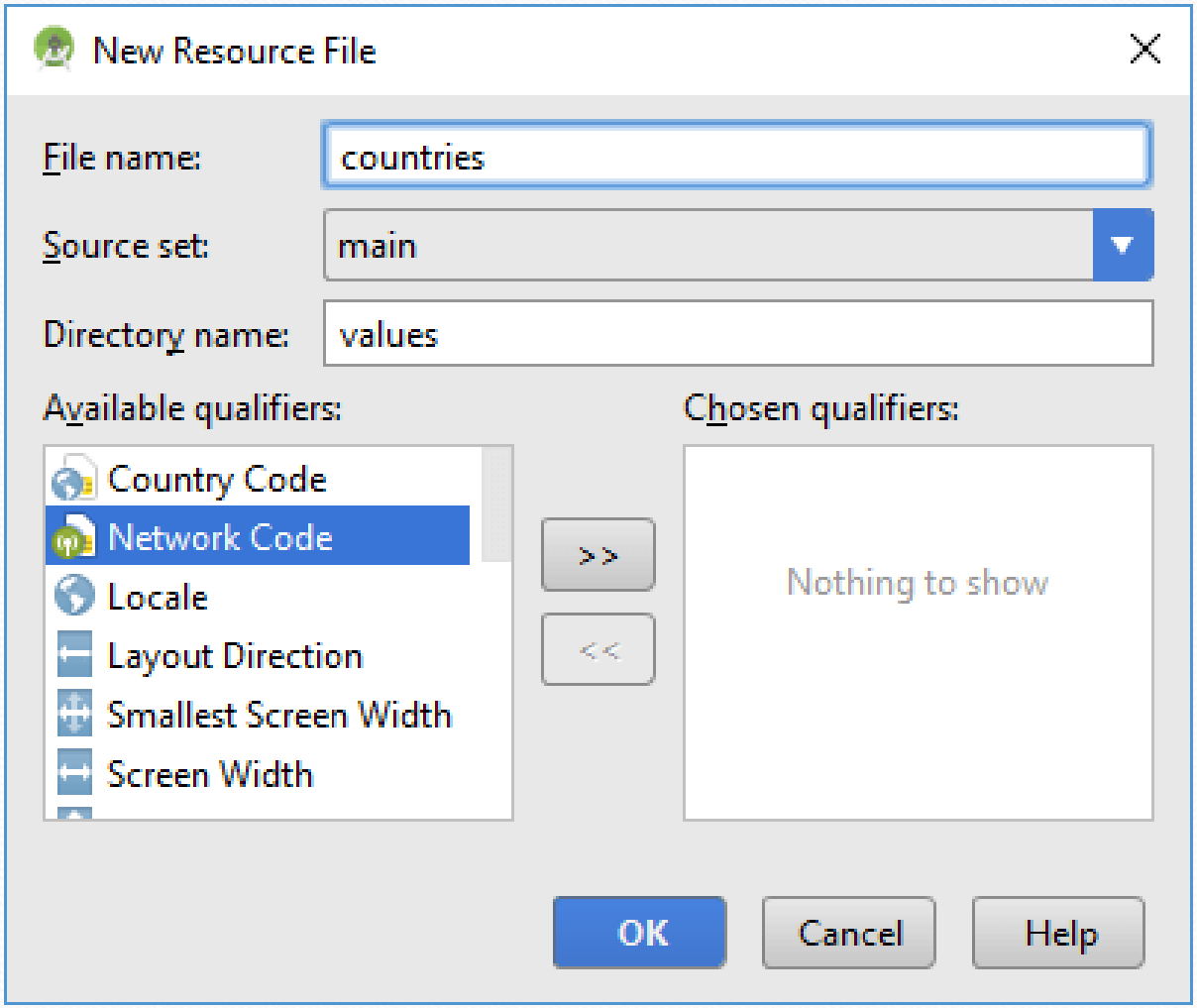
Мы рассмотрели, как выводить массив строк с помощью ArrayAdapter в ListView. При этом массив строк определялся программно в коде java. Однако подобный массив строк можно было бы хранить в файле xml в виде ресурса.

Ресурсы массивов строк представляют элемент типа **string-array**. Они могут находится в каталоге *res/values* в xml-файле с произвольным именем.

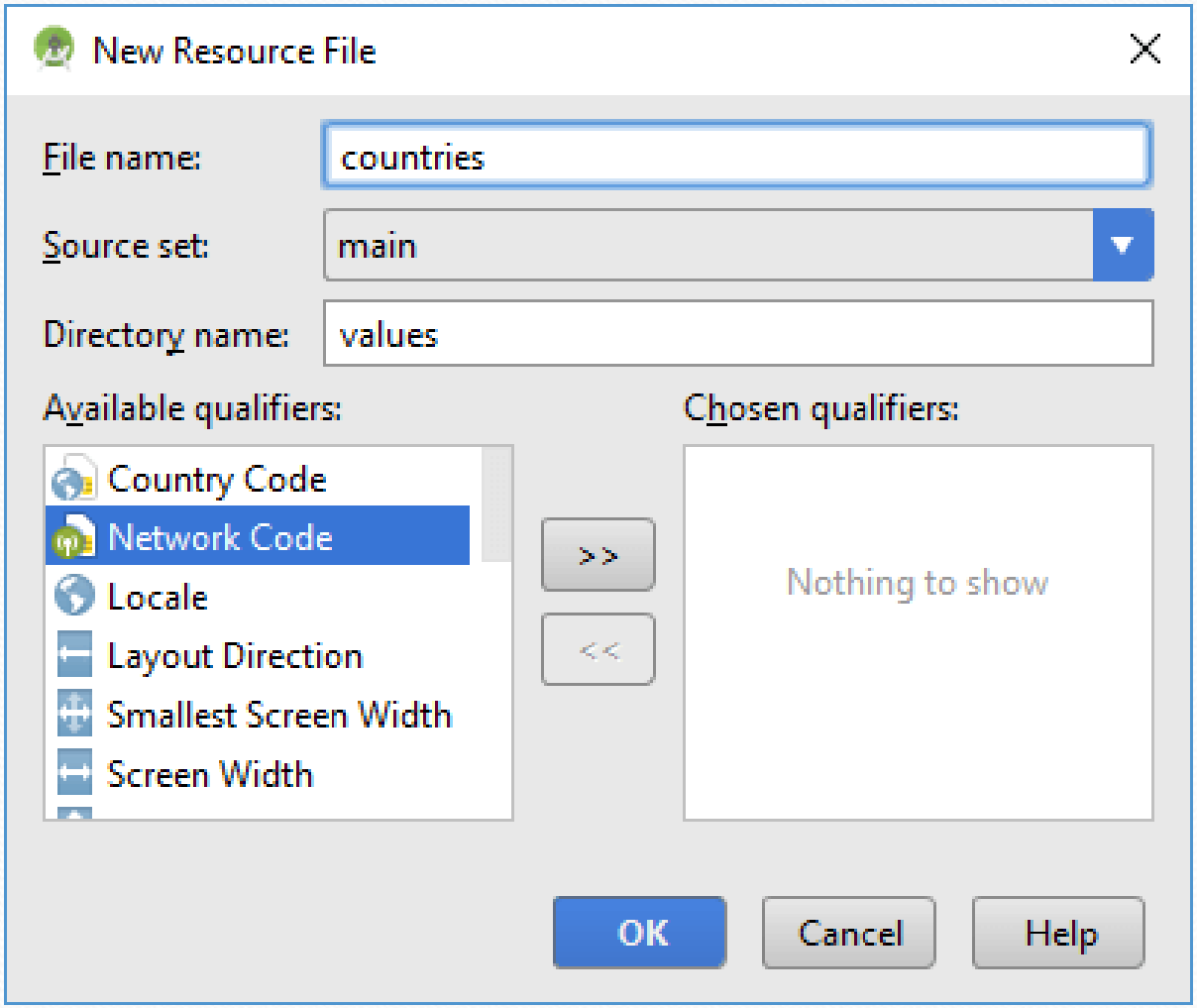
Создаем новый XML-файл



Создаем новый XML-файл



Создаем новый XML-файл



string-array

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <resources>
3      <string-array name="имя_массива_строк">
4          <item>элемент</item>
5      </string-array>
6  </resources>
```

Заполняем ресурс

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="countries">
        <item>Бразилия</item>
        <item>Аргентина</item>
        <item>Колумбия</item>
        <item>Чили</item>
        <item>Уругвай</item>
    </string-array>
</resources>
```

Используем ресурс

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

    // получаем элемент ListView
    ListView countriesList = (ListView) findViewById(R.id.countriesList);

    // получаем ресурс
    String[] countries = getResources().getStringArray(R.array.countries);

    // создаем адаптер
    ArrayAdapter<String> adapter = new ArrayAdapter(this,
            android.R.layout.simple_list_item_1, countries);

    // устанавливаем для списка адаптер
    countriesList.setAdapter(adapter);
}
```

Атрибут **entries**

Необязательно добавлять список строк в ListView программно. У этого элемента есть атрибут **entries**, который может принимать ресурс string-array:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ListView
        android:entries="@array/countries"
        android:id="@+id/countriesList"
        android:layout_width="match_parent"
        android:layout_height="match_parent">
    </ListView>

</RelativeLayout>
```

Минимальный код

В этом случае код MainActivity мы можем сократить до минимума:

```
3  import android.support.v7.app.AppCompatActivity;
4  import android.os.Bundle;
5
6  public class MainActivity extends AppCompatActivity {
7
8      @Override
9      protected void onCreate(Bundle savedInstanceState) {
10         super.onCreate(savedInstanceState);
11         setContentView(R.layout.activity_main);
12     }
13 }
```

Обработка нажатий

Для обработки выбора элемента списка используется слушатель **OnItemClickListener**. Этот слушатель имеет один метод **onItemClick**, через параметры которого мы можем получить выделенный элемент и сопутствующие данные. Так, он принимает следующие параметры:

- **parent** : нажатый элемент AdapterView (в роли которого в данном случае выступает наш элемент ListView)
- **view** : нажатый виджет внутри AdapterView
- **position** : индекс нажатого виждета внутри AdapterView
- **id** : идентификатор строки нажатого элемента

Обработка нажатий списка

```
// добвляем для списка слушатель
countriesList.setOnItemClickListener(new OnItemClickListener(){
    @Override
    public void onItemClick(AdapterView<?> parent, View v, int position, long id)
    {
        // по позиции получаем выбранный элемент
        String selectedItem = countries[position];
        // установка текста элемента TextView
        selection.setText(selectedItem);
    }
});
```

Множественный выбор

Иногда требуется выбрать не один элемент, как по умолчанию, а несколько. Для этого, во-первых, в разметке списка надо установить атрибут

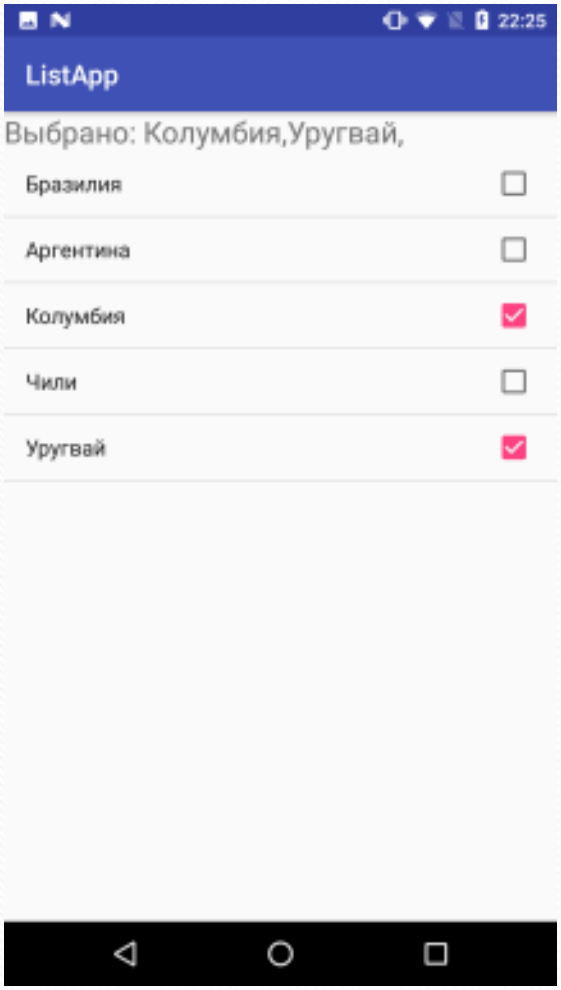
android:choiceMode="multipleChoice"

```
<ListView
    android:choiceMode="multipleChoice"
    android:id="@+id/countriesList"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

```
ArrayAdapter<String> adapter = new ArrayAdapter(this,
        android.R.layout.simple_list_item_multiple_choice, countries);
// устанавливаем для списка адаптер
countriesList.setAdapter(adapter);
// добвляем для списка слушатель
countriesList.setOnItemClickListener(new OnItemClickListener(){
    @Override
    public void onItemClick(AdapterView<?> parent, View v, int position, long id)
    {
        SparseBooleanArray sp=countriesList.getCheckedItemPositions();

        String selectedItems="";
        for(int i=0;i < countries.length;i++)
        {
            if(sp.get(i))
                selectedItems+=countries[i]+",";
        }
        // по позиции получаем выбранный элемент
        String selectedItem = countries[position];
        // установка текста элемента TextView
        selection.setText("Выбрано: " + selectedItems);
```

Множественный выбор

При выборе элементов мы получаем все выбранные позиции в объект **SparseBooleanArray**, затем пробегаемся по всему массиву, и если позиция элемента в массиве есть в SparseBooleanArray, то есть она отмечена, то добавляем отмеченный элемент в строку.



Адаптер. Работа со списком

После привязки ListView к источнику данных через адаптер можно работать с данными - добавлять, удалять, изменять только через адаптер. ListView служит только для отображения данных.

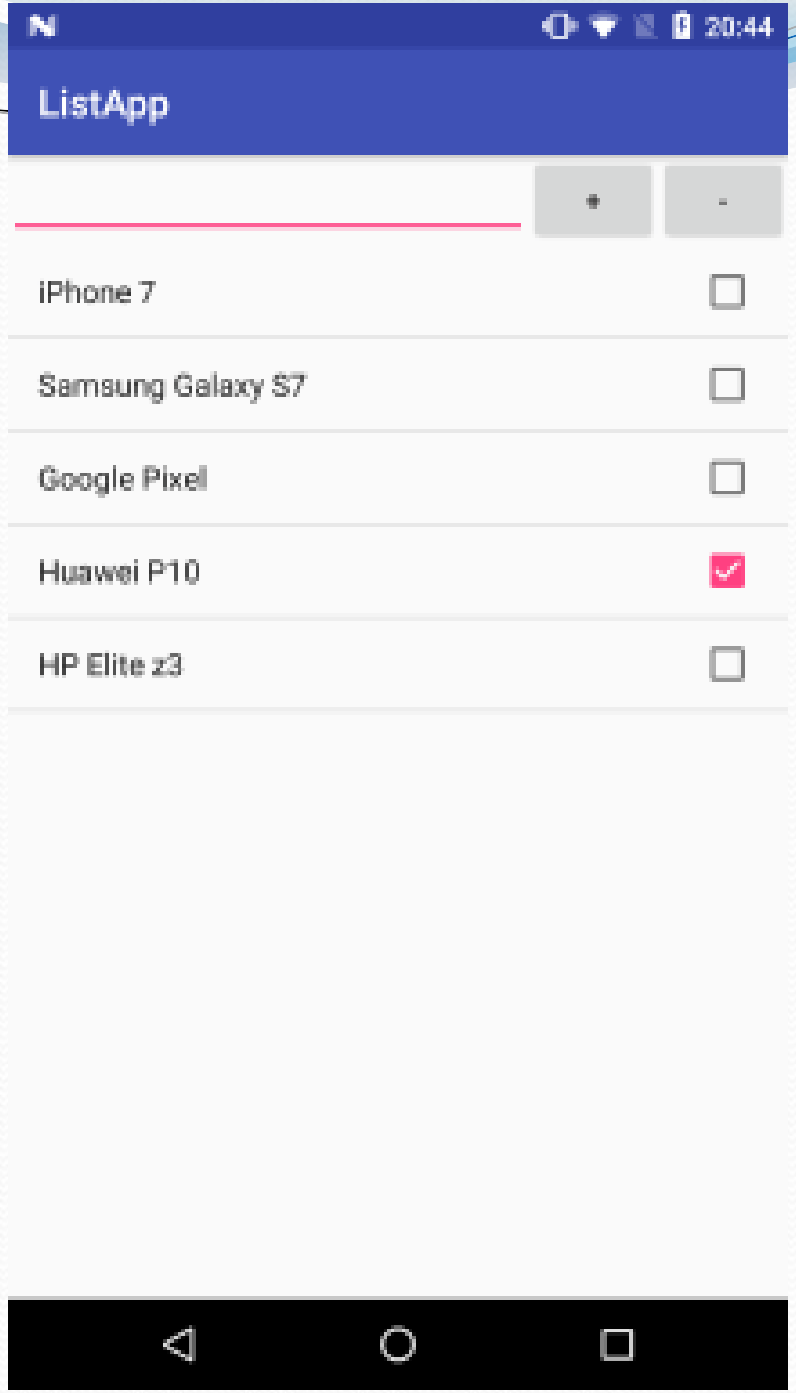
Для управления данными мы можем использовать методы адаптера или напрямую источника данных. Например, с помощью метода **add**() класса **ArrayAdapter** можно добавить новый элемент в конец массива-источника данных.

Однако после применения вышеуказанных методов изменения коснутся только массива, выступающего источником данных. Чтобы синхронизировать изменения с элементом ListView, надо вызвать у адаптера метод **notifyDataSetChanged**().

Функции ArrayAdapter

- add() - позволяет добавить новый элемент в конец массива-источника данных
- insert() - позволяет добавить новое значение по определенному индексу
- remove() - позволяет удалить объект из массива
- sort() - позволяет провести сортировку массива

Пример


20:44
ListApp
+
-
iPhone 7
Samsung Galaxy S7
Google Pixel
Huawei P10
HP Elite z3

Создаем список

```
public class MainActivity extends AppCompatActivity {

    ArrayList<String> phones = new ArrayList();
    ArrayAdapter<String> adapter;

    ArrayList<String> selectedPhones = new ArrayList();
    ListView phonesList;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        phones.add("iPhone 7");
        phones.add("Samsung Galaxy S7");
        phones.add("Google Pixel");
        phones.add("Huawei P10");
        phones.add("HP Elite z3");

        phonesList = (ListView) findViewById(R.id.phonesList);
        adapter = new ArrayAdapter<String>(this, android.R.layo
        phonesList.setAdapter(adapter);
```

Обрабатываем нажатия

```
// обработка установки и снятия отметки в списке
phonesList.setOnItemClickListener(new AdapterView.OnItemClickListener(){
    @Override
    public void onItemClick(AdapterView<?> parent, View v, int position, long id)
    {
        // получаем нажатый элемент
        String phone = adapter.getItem(position);
        if(phonesList.isItemChecked(position)==true){
            selectedPhones.add(phone);
        }
        else{

            selectedPhones.remove(phone);
        }
    }
});
```

Добавление в список

```
public void add(View view){

    EditText phoneEditText = (EditText) findViewById(R.id.phone);
    String phone = phoneEditText.getText().toString();
    if(!phone.isEmpty() && phones.contains(phone)==false){
        adapter.add(phone);
        phoneEditText.setText("");
        adapter.notifyDataSetChanged();
    }
}
```

Удаление из списка

```
public void remove(View view){
    // получаем и удаляем выделенные элементы
    for(int i=0; i< selectedPhones.size();i++){
        adapter.remove(selectedPhones.get(i));
    }
    // снимаем все ранее установленные отметки
    phonesList.clearChoices();
    // очищаем массив выбраных объектов
    selectedPhones.clear();

    adapter.notifyDataSetChanged();
}
```