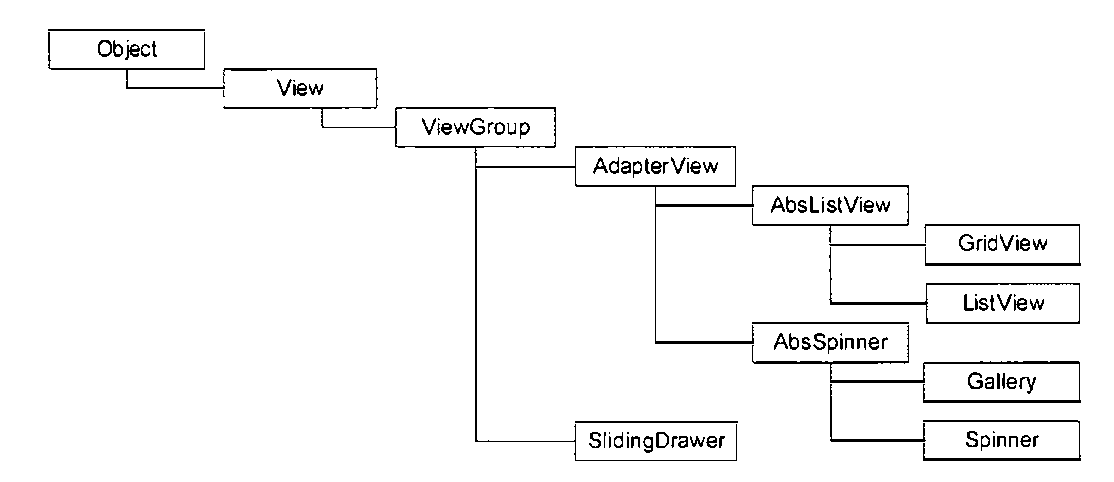
# Отображение данных в списках

### Элементы-списки в Android представляют пять классов:

1. ListView;
2. GridView;
3. Gallery;
4. Spinner;
5. SlidingDrawer.

Это контейнерные виджеты, которые (кроме SlidingDrawer) являются подклассами AdapterView. Эти виджеты можно использовать для связывания с определенным типом данных и отображения их пользователю. Иерархия классов списков представлена на рисунке.



Иерархия классов списков

**Класс AdapterView** предоставляет две основные функциональности для работы со списками:

* заполнение схемы размещения с данными;
* обработку выбора элемента данных пользователем.

**AdapterView—подкласс ViewGroup**, дочерние представления которого определены**объектом Adapter**, который связывает их с данными некоторого типа.

Объект Adapter действует подобно курьеру между вашим источником данных (например, массивом строк) и AdapterView, который отображает эти данные.

Есть несколько реализаций класса Adapter для определенных задач, например CursorAdapter для чтения данных из объекта Cursor или ArrayAdapter для чтения из произвольного массива. Иногда необходимо заполнить группу представлений небольшим количеством информации, которая не может быть жестко закодирована, а будет связана с внешним источником данных, например базой данных SQLite. Чтобы сделать это, необходимо использовать объект AdapterView как группу представлений и каждый дочерний View инициализировать и заполнять данными от объекта Adapter.

**Класс AdapterView** является базовым для **класса AbsListView**, который может использоваться для реализации классов виджетов, представляющих списки и таблицы (**ListView и GridView**), и **класса AbsSpinner**— для выпадающих списков и галереи с прокруткой (Gallery и Spinner).

* [GridView](http://androidfanclub.ru/programming/gridview)
* [ListView](http://androidfanclub.ru/programming/listview)
* [Spinner](http://androidfanclub.ru/programming/spinner)

**GridView**

**Виджет GridView** представляет собой плоскую таблицу. Для **GridView**, вместо того чтобы использовать автоматически генерируемые **виджеты TextView**, как в случае с **элементом ListView**, можно использовать собственные поля для отображения элементов данных, создав класс, производный от **класса ArrayAdapter**, и переопределив его **метод getView ().**

Число столбцов для **GridView** чаще задается статически. Число строк в элементе определяется динамически на основании числа элементов, которые предоставляет адаптер.

Есть несколько свойств, которые определяют число столбцов и их размеры:

* **android:numColumns**— определяет количество столбцов. Если поставлено значение autofit, то система вычислит количество столбцов, основанное на доступном пространстве;
* **android: verticalSpacing**— устанавливает размер пустого пространства между ячейками таблицы;
* **android: columnWidth** — устанавливает ширину столбцов;
* **android: stretchMode**— указывает, куда распределяется остаток свободного пространства для таблицы с установленным значением android: numColumns="auto\_fit". Принимает значения columnWidth для распределения остатка свободного пространства между ячейками столбцы для их увеличения или spacingWidth — для увеличения пространства между ячейками.

В качестве примера использования элемента GridView в приложении для отображения текстовой информации создайте в **Eclipse** новый проект и в диалоге *Create New Project*введите следующие значения:

* Project name — GridViewApp;
* Application name — GridView Sample;
* Package name — com. samples. gridview;
* Create Activity — GridViewActivi ty.

Откройте файл разметки и создайте структуру разметки подобно листингу (*файл разметки main.xml):*

<?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">

<TextView

android:id="@+id/label"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"/>

<GridView

android:id="@+id/grid"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:verticalSpacing="35px"

android:horizontalSpacing="5px"

android:numColumns="auto\_fit"

android:columnWidth="100px"

android:stretchMode="columnWidth"

android:gravity="center"/>

</LinearLayout>

Код класса оболочки для **данных DataAdapter**, производного от **ArrayAdapter**, представлен в листинге (*файл класса адаптера данных DataAdapter.java*):

package com.samples.gridview;

import android.content.Context;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ArrayAdapter;

import android.widget.TextView;

public class DataAdapter extends ArrayAdapter<String>

{

private static final String[] mContacts = {

"J. Anderson", "E. Duncan", "M. Fuller",

"E. Greenman", "J. Harrison", "M. Johnson",

"M. Cotman", "O. Lawson", "A. Chapman",

"M. Honeyman", "I. Jackson", "W. Patterson",

"J. Godwin", "S. Bush", "C. Gateman",

"J. Anderson", "E. Duncan", "M. Fuller",

"E. Greenman", "J. Harrison", "M. Johnson",

"M. Cotman", "O. Lawson", "A. Chapman",

"M. Honeyman", "I. Jackson", "W. Patterson",

"J. Godwin", "S. Bush", "C. Gateman" };

Context mContext;

DataAdapter(Context context, int resource) {

super(context, resource, mContacts);

this.mContext = context;

}

public View getView(int position, View convertView, ViewGroup parent) {

TextView label = (TextView)convertView;

if (convertView == null) {

convertView = new TextView(mContext);

label = (TextView)convertView;

}

label.setText(mContacts[position]);

return(convertView);

}

public String GetItem(int position) {

return mContacts[position];

}

}

В целом, **GridView** работает так же, как и любой другой элемент из рассмотренных ранее: для связывания данных и дочерних представлений используется **метод setAdapter()**, для регистрации слушателя события выбора ячейки вызывается **setOnitemSelectedListener(),** и в коде класса деятельности реализуются **методы onitemSelected () и onNothingSelected ().**

*Файл класса деятельности GridViewActivity.Java*

package com.samples.gridview;

import android.app.Activity;

import android.os.Bundle;

import android.view.View;

import android.widget.AdapterView;

import android.widget.GridView;

import android.widget.TextView;

public class GridViewActivity extends Activity implements AdapterView.OnItemSelectedListener

{

private TextView mSelectText;

DataAdapter mAdapter;

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

mSelectText=(TextView)findViewById(R.id.label);

final GridView grid = (GridView)findViewById(R.id.grid);

mAdapter = new DataAdapter(getApplicationContext(), android.R.layout.simple\_list\_item\_1);

grid.setAdapter(mAdapter);

grid.setOnItemSelectedListener(this);

}

@Override

public void onItemSelected(AdapterView<?> parent, View v, int position, long id) {

mSelectText.setText("Selected items: " + mAdapter.GetItem(position));

}

@Override

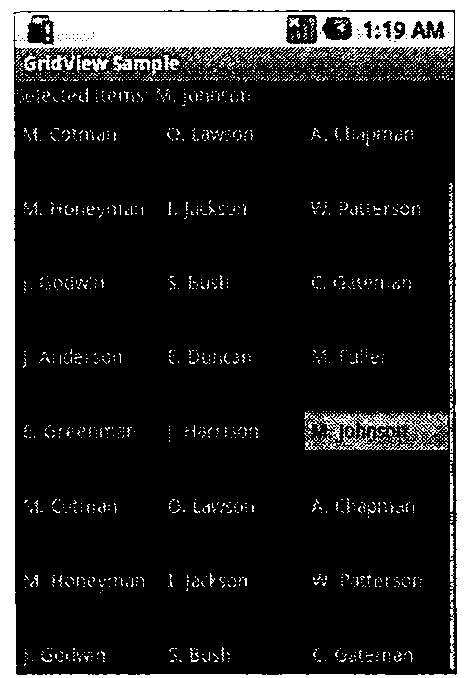
public void onNothingSelected(AdapterView<?> parent) {

mSelectText.setText("Selected items: none");

}

}

Запустите проект на выполнение. При выборе элемента списка в окне текстового поля вверху экрана будет отображаться содержимое выбранного элемента:



*Приложение с виджетом****GridView***

**ListView**

**Элемент ListView**представляет собой вертикальный список с прокруткой. Связанные со списком данные ListView получает от **объекта ListAdapter.** В отличие от предыдущих примеров, когда мы использовали в качестве базового класса **Activity**, при работе с **ListView**в качестве базового применяется **класс ListActivity.**

**Класс ListActivity** реализует отображение списка элементов, привязанных к источнику данных, например к массиву, и набор методов обратного вызова для обработки событий выбора элементов списка данных, т. е. **ListActivity** является хостом для **объекта ListView**, который может быть связан с различными источниками данных. У **ListActivity** есть заданная по умолчанию разметка, которая состоит из единственного списка, растянутого на весь экран (точнее — на родительский контейнер).

*Для связывания объекта ListActivity с данными необходимо разработать класс, который реализует интерфейс ListAdapter. Android обеспечивает два стандартных адаптера списка:*

* SimpleAdapter;
* SimpleCursorAdapter.

**SimpleAdapter** применяется для статического связывания данных небольшого объема.**SimpleCursorAdapter** используется, как правило, при формировании выборки из больших массивов данных.

В качестве примера использования **элемента Listview** в приложении создайте новый проект и в диалоге *Create New Project* введите следующие значения:

* Project name — ListViewApp;
* Application name — ListView Sample;
* Package name — сот. samples .listview4;
* Create Activity — ListViewActivity.

Откройте файл разметки и создайте структуру разметки с контейнером LinearLayout и вложенными виджетами ListView для отображения списка и TextView для отслеживания события выбора элемента списка, как в листинге (*файл разметки main.xml*):

<?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">

<TextView

android:id="@+id/textSelect"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:textStyle="bold"/>

<ListView

android:id="@android:id/list"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:drawSelectorOnTop="false"/>

</LinearLayout>

У **ListActivity** есть заданная по умолчанию разметка, которая состоит из единственного списка, растянутого на весь экран.

*Файл класса деятельности ListViewActivity.java*

package com.samples.listview;

import android.app.ListActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.ArrayAdapter;

import android.widget.ListView;

import android.widget.TextView;

public class ListViewActivity extends ListActivity {

private TextView mTextView;

String[] mContacts = {

"Jacob Anderson", "Emily Duncan", "Michael Fuller",

"Emma Greenman", "Joshua Harrison", "Madison Johnson",

"Matthew Cotman", "Olivia Lawson", "Andrew Chapman",

"Daniel Honeyman", "Isabella Jackson", "William Patterson",

"Joseph Godwin", "Samantha Bush", "Christopher Gateman"};

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

setListAdapter(new ArrayAdapter<String>(this, android.R.layout.simple\_list\_item\_1, mContacts));

mTextView = (TextView)findViewById(R.id.textSelect);

}

public void onListItemClick(

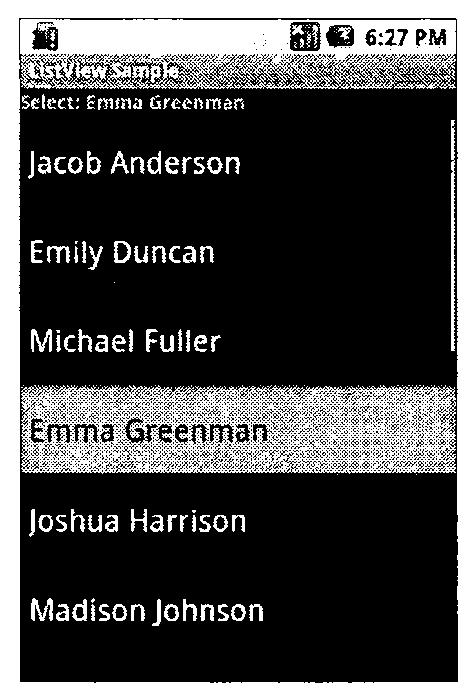
ListView parent, View v, int position, long id) {

mTextView.setText("Select: " + mContacts[position] + " " + position + " " + id);

}

}

Запустите проект на выполнение. При выборе элемента списка в окне текстового поля вверху окна деятельности будет отображаться содержимое выбранного элемента.



*Приложение с виджетом ListView*

**Spinner**

**Виджет Spinner**— это ***аналог ComboВох*** (выпадающий список) для Android. При нажатии кнопки на элементе или центральной кнопки на Dклавиатуре устройства появляется список опций с радиокнопками, который, в отличие от элемента ListView, не занимает весь экран.

Основное событие элемента Spinner, которое реализуют в программе, использующей Spinner, — это событие выбора пункта списка. Для этого в коде программы необходимо реализовать методы обратного вызова  ***onltemSelected () и onNothingSelected ()***, которые объявлены в**интерфейсе AdapterView.OnltemSelectedListener**.

Для примера приложения с использованием выпадающего списка создайте новый проект и в диалоге *Create New Project* введите следующие значения:

* Project name — SpinnerApp;
* Application name — Spinner Sample;
* Package name —сот. samples. spinner;
* Create Activity — SpinnerActivity.

Откройте файл разметки и создайте структуру разметки подобно листингу (*файл разметки main.xml*):

<?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">

<TextView

android:id="@+id/TextView01"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content">

</TextView>

<Spinner

android:id="@+id/Spinner01"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:drawSelectorOnTop="true">

</Spinner>

</LinearLayout>

Полный код **класса SpinnerActivity (***файл класса деятельности SpinnerActivity.Java*):

package com.samples.spinner;

import com.samples.spinner.R;

import android.app.Activity;

import android.os.Bundle;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.Spinner;

import android.widget.TextView;

public class SpinnerActivity extends Activity

implements AdapterView.OnItemSelectedListener {

TextView mLabel;

final String[] mContacts = {

"Jacob Anderson", "Emily Duncan", "Michael Fuller",

"Emma Greenman", "Joshua Harrison", "Madison Johnson",

"Matthew Cotman", "Olivia Lawson", "Andrew Chapman",

"Michael Honeyman", "Isabella Jackson", "William Patterson",

"Joseph Godwin", "Samantha Bush", "Christopher Gateman"};

@Override

public void onCreate(Bundle icicle) {

super.onCreate(icicle);

setContentView(R.layout.main);

mLabel = (TextView)findViewById(R.id.TextView01);

final Spinner spin = (Spinner)findViewById(R.id.Spinner01);

spin.setOnItemSelectedListener(this);

ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(

this, android.R.layout.simple\_spinner\_item, mContacts);

arrayAdapter.setDropDownViewResource(

android.R.layout.simple\_spinner\_dropdown\_item);

spin.setAdapter(arrayAdapter);

}

public void onItemSelected(

AdapterView<?> parent, View v, int position, long id) {

mLabel.setText(mContacts[position]);

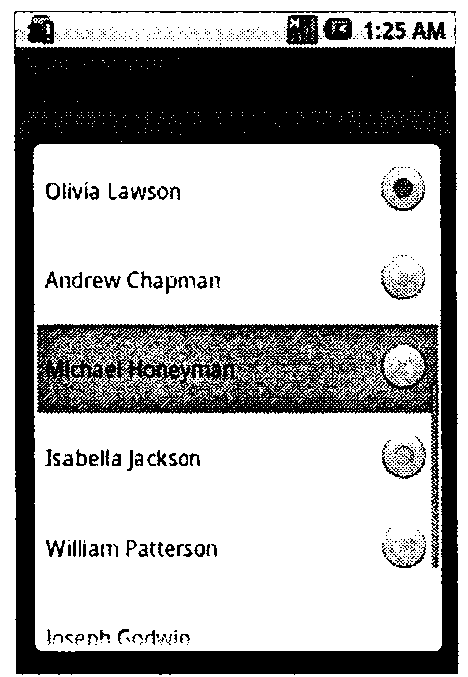
}

public void onNothingSelected(AdapterView<?> parent) {

mLabel.setText("");

}

}



*Приложение с виджетом Spinner*

Запустите проект на выполнение. Внешний вид приложения с виджетом Spinner и открытым выпадающим списком представлен на рисунке.

**Создание списка с собственной разметкой**

Кроме использования стандартных виджетовсписков можно создать список, определив собственную разметку только для одной строки этого списка. Такой подход используется для связывания данных, представленных в виде плоских таблиц. Для связывания табличных данных используются классы **SimpleAdapter И SimpleCursorAdapter.**

**SimpleAdapter**— адаптер для связывания статических данных с представлением, определенным в XMLфайле. Конструктор класса выглядит так:

SimpleAdapter (Context context, List<? extends Map<String,

?>> data, int resource, String [] from, int [] to)

Данные представляются как список объектов мар, которые, в свою очередь, содержат данные для каждой строки: множества элементов ключзначение, где ключ — это имя поля, значение — содержимое поля. То есть каждый элемент в ArrayList соответствует одной строке данных в списке.

**Класс SimpleCursorAdapter** применяют для связывания представлений с базами данных.

*В качестве примера разработаем приложение, отображающее статические данные в табличном виде. Создайте новый проект и в диалоге Create New Project введите следующие значения:*

* Project name — ListContact;
* Application name — Contacts Sample;
* Package name — соm.samples .listcontact;
* Create Activity — ListContactActivity.

В приложении будем отображать список контактов, приведенный ранее, но в строке будет два столбца: поле *Name*с выравниванием влево и поле *Phone* с выравниванием вправо.

Для этого в файле разметки определим строку с двумя элементами **TextView**, как показано в листинге:

Листинг. Файл разметки main.xml

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

<RelativeLayout 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/name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentLeft="true"

android:textSize="18sp"/>

<TextView

android:id="@+id/phone"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentRight="true"

android:textSize="18sp"

android:paddingRight="10px"/>

</RelativeLayout>

Для хранения строки, представляющей контакт, создадим отдельный класс contactitem, расширяющий класс наэпмар, в котором определим константы *name и phone* с именами полей и конструктор с параметрами — имя контакта и телефон. Код класса представлен в листинге:

Листинг. Класс Contactitem

package com.samples.listcontact;

import java.util.HashMap;

public class ContactItem extends HashMap<String, String> {

private static final long serialVersionUID = 1L;

public static final String NAME = "name";

public static final String PHONE = "phone";

public ContactItem(String name, String phone) {

super();

super.put(NAME, name);

super.put(PHONE, phone);

}

}

В классе деятельности**ListContactActivity** заполним данными **объект ArrayList** и отобразим его в виде таблицы в окне деятельности:

Листинг. Класс деятельности ListContactActivity

package com.samples.listcontact;

import java.util.ArrayList;

import android.app.ListActivity;

import android.os.Bundle;

import android.widget.ListAdapter;

import android.widget.SimpleAdapter;

public class ListContactActivity extends ListActivity {

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

ArrayList<ContactItem> list = new ArrayList<ContactItem>();

list.add(new ContactItem("Jacob Anderson", "412412411"));

list.add(new ContactItem("Emily Duncan", "161863187"));

list.add(new ContactItem("Michael Fuller", "896443658"));

list.add(new ContactItem("Emma Greenman", "964990543"));

list.add(new ContactItem("Joshua Harrison", "759285086"));

list.add(new ContactItem("Madison Johnson", "950285777"));

list.add(new ContactItem("Matthew Cotman", "687699999"));

list.add(new ContactItem("Olivia Lawson", "161863187"));

list.add(new ContactItem("Andrew Chapman", "546599645"));

list.add(new ContactItem("Daniel Honeyman", "876545644"));

list.add(new ContactItem("Isabella Jackson", "907868756"));

list.add(new ContactItem("William Patterson", "687699693"));

list.add(new ContactItem("Joseph Godwin", "965467575"));

list.add(new ContactItem("Samantha Bush", "907865645"));

list.add(new ContactItem("Christopher Gateman", "896874556"));

ListAdapter adapter = new SimpleAdapter(this, list, R.layout.main,

new String[] {ContactItem.NAME, ContactItem.PHONE},

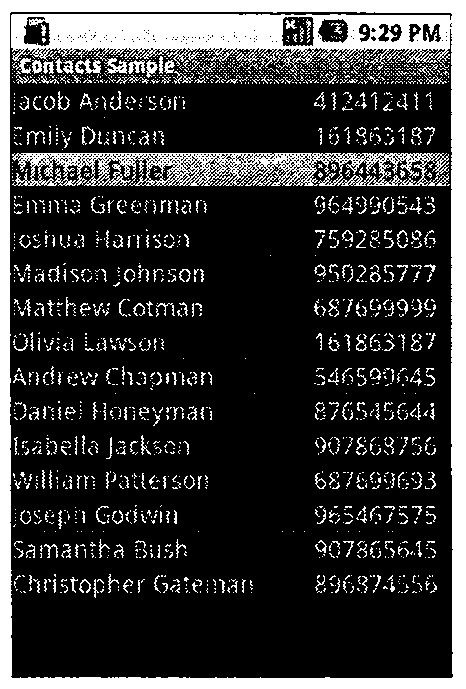
new int[] {R.id.name, R.id.phone});

setListAdapter(adapter);

}

}

Внешний вид приложения со списком контактов представлен на рисунке:



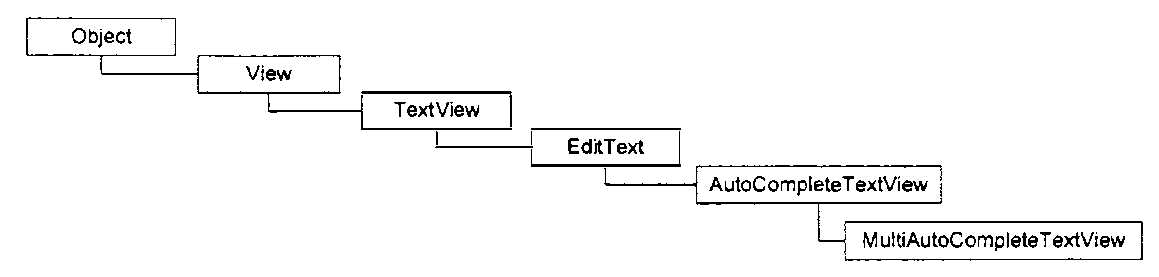
*Список с собственной разметкой*

**Текстовые поля с автозаполнением**

Текстовые поля с автозаполнением в Android представлены двумя классами:

* **AutoCornpleteTextView;**
* **MultiAutoCornpleteTextView.**

Эти классы наследуют все методы для работы с текстом от **класса TextView** и редактирования текста от **класса EditText**. Иерархия классов текстовых полей с автозаполнением представлена на рисунке:



*Иерархия классов текстовых полей с автозаполнением*

* [AutoCompleteTextView](http://androidfanclub.ru/programming/autocompletetextview)
* [MultiAutoCompleteTextView](http://androidfanclub.ru/programming/multiautocompletetextview)