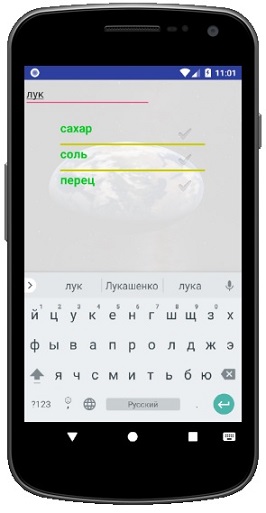
**Настройка интерфейса ListView**

Данная статья продолжает описание прокручиваемого списка [ListView](http://java-online.ru/android-listview.xhtml). В предыдущей статье в примерах были использованы системные разметки компонента *simple\_list\_item\_1* и *simple\_list\_item\_checked*. Использование системной разметки хорошо тем, что нет необходимости разрабатывать*layout* компонента ListView. Однако, если имеется желание изменить интерфейс списка, то можно создать свой *layout*-ресурс, что и будет рассмотрено в данной статье с использованием примера, описанного в предыдущей статье. Конечно же, изменения в примере будут существенными, и, в первую очередь, они будут касаться прокручиваемого списка. В статье будут рассмотрены :

* настройка интерфейса прокручиваемого списка;
* динамическое изменение списка.

Таким образом, в примере будет не только определен собсвенный интерфейс прокручиваемого списка *ListView*, но и представлена возможность динамически добавлять элементы в список. На следующем скриншоте изображен интерфейс запущенного примера, описание которого представлено ниже. В верхней части размещен компонент ввода текстового значения, которое используется для добавления нового элемента в список. Ниже размещен прокручиваемый список на фоне земли. В нижней части открыта виртуальная клавиатура.



***Примечание :****для подключения дополнительного/русского языка в виртуальной клавиатуре необходимо в виртуальном устройстве открыть настройки и добавить свой/русский язык. После этого в виртуальной клавиатуре можно будет выбирать язык.*

**Пример настройки прокручиваемого списка**

При создании собственного интерфейса разметки прокручиваемого списка **ListView** необходимо определиться с функционалом компонента, чтобы правильно выбрать шаблон интерфейса. Допустим, что нам необходимо в прокручиваемом списке выделять несколько записей, подобно тому, как это происходит в настройках какой-либо игры. То есть, необходимо использовать список с *checkbox'ами*. Для этого в качестве основы нашей разметки [используем](http://java-online.ru/android-listview.xhtml#listview-layout) *android.R.layout.simple\_list\_item\_checked* и создадим собственный шаблон *list\_item.xml* :

**Разметка списка list\_item.xml**

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

<CheckedTextView

xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@+id/list\_item"

android:layout\_width="220dp"

android:layout\_height="40dp"

android:textColor="#00cc22"

android:textSize="20sp"

android:textStyle="bold"

android:checkMark="?android:attr/textCheckMark">

</CheckedTextView>

В разметке компонента определяются параметры интерфейса отдельной записи списка *ListView* :

* идентификатор записи android:id;
* размер записи android:layout\_width, android:layout\_height;
* цвет записи android:textColor;
* размер и стиль шрифта android:textSize, android:textStyle;
* наличие чекбокса android:checkMark.

**Описание интерфейса**

В примере используется разделитель элементов списка. Разделитель определяется в описании интерфейсной разметки (layout) компонента. В примере разделителем является обычная линия, цвет которой (name="color\_yellow") необходимо определить в ресурсном файле *res/color.xml*.

**Ресурсный файл color.xml**

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

<resources>

<color name="color\_yellow">

#cccc00

</color>

</resources>

*Если в качестве разделителя использовать ресурсное изображение, например res/drawable/divider.png, то необходимо учитывать наличие фонового изображения, т.е. "прозрачность" разделителя.*

**Листинг разметки**

В описании разметки интерфейса присутствуют два компонента : EditText, ListView. В качестве фонового изображения используются описанные в первой части статьи изображения @drawable/earth\_portrait и @drawable/earth\_landscape. Интерес, с точки зрения темы статьи, представляет описание компонента *ListView*, включающее :

* атрибут выбора нескольких компонентов choiceMode;
* атрибуты позиционирования paddingTop и paddingLeft;
* разделитель divider (может быть использовано изображение);
* атрибут размера разделителя по высоте dividerHeight.

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

<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:id="@+id/mainlayout"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:background="@drawable/earth\_portrait"

tools:context=".MainActivity">

<EditText

android:id="@+id/etItem"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="" />

<ListView

android:id="@+id/list"

android:choiceMode="multipleChoice"

android:paddingTop="20dp"

android:paddingLeft="60dp"

android:divider="@color/color\_yellow"

android:dividerHeight="3dp"

android:layout\_width="300dp"

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

</ListView>

</LinearLayout>

**Листинг активности приложения**

В коде активности изменился только метод onCreate. Методы onResume, onConfigurationChanged, setBackgroundImage описаны в предыдущей статье.

На что следует обратить внимание в методе onCreate?

1. В качестве списка ресурсных значений *list* используется тип List<String>. Обратите внимание на то, как формируется набор данных. Если просто использовать *list = Arrays.asList(data)*, то невозможно будет [добавить](http://java-online.ru/java-arrayList.xhtml#arrays) в набор новое значение.
2. При формирование адптера данных списка используется разметка *R.layout.list\_item* и набор *list*.
3. К компоненту ввода текстовых значений *editText* подключен слушатель, который после ввода с клавиатуры KeyEvent.KEYCODE\_ENTER добавляет значение в набор данных, вызывает метод обновления данных в списке и обнуляет поле ввода.

import java.util.List;

import java.util.Arrays;

import java.util.ArrayList;

import android.os.Bundle;

import android.view.View;

import android.view.KeyEvent;

import android.app.Activity;

import android.widget.EditText;

import android.widget.ListView;

import android.widget.ArrayAdapter;

import android.widget.LinearLayout;

import android.content.res.Configuration;

import android.graphics.drawable.Drawable;

public class MainActivity extends Activity

{

private ArrayAdapter<String> adapter ;

private List<String> list ;

private LinearLayout linearLayout;

private int orientation ;

@Override

protected void onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_custom);

linearLayout = findViewById(R.id.mainlayout);

orientation = Configuration.ORIENTATION\_PORTRAIT;

String[] data;

data = getResources().getStringArray(R.array.names);

list = new ArrayList<>(Arrays.asList(data));

adapter = new ArrayAdapter<>(this,

R.layout.list\_item, list);

ListView listView = findViewById(R.id.list);

listView.setAdapter(adapter);

final EditText editText = findViewById(R.id.etItem);

// Подключаем слушателя клавиш

editText.setOnKeyListener(new View.OnKeyListener() {

public boolean onKey(View v, int keyCode,

KeyEvent event) {

if (event.getAction()==KeyEvent.ACTION\_DOWN)

if (keyCode == KeyEvent.KEYCODE\_ENTER) {

String txt;

txt = editText.getText().toString();

list.add(txt);

adapter.notifyDataSetChanged();

editText.setText("");

return true;

}

return false;

}

});

}

@Override

public void onResume()

{

super.onResume();

setBackgroundImage(orientation);

}

@Override

public void onConfigurationChanged(Configuration config)

{

super.onConfigurationChanged(config);

setBackgroundImage(config.orientation);

orientation = config.orientation;

}

private void setBackgroundImage(final int orientation)

{

Drawable image;

image = this.getResources().getDrawable(

R.drawable.earth\_landscape, null);

if (orientation==Configuration.ORIENTATION\_PORTRAIT)

image = this.getResources().getDrawable(

R.drawable.earth\_portrait, null);

image.setAlpha(32);

linearLayout.setBackground(image);

}

}