**LinearLayout**

**Разметка LinearLayout** выравнивает все дочерние объекты представлений в одном направлении — вертикально или горизонтально, в зависимости от того, как определен атрибут ориентации *android:orientation:*

android:orientation="horizontal"

или

android:orientation="vertical"

Все дочерние записи помещаются в стек один за другим, так что вертикальный список представлений будет иметь только один дочерний элемент в строке независимо от того, насколько широким он является. Горизонтальное расположение списка будет размещать элементы в одну строку с высотой, равной высоте самого высокого дочернего элемента списка.

Для примера окна с линейной разметкой создайте новый проект и в диалоговом окне New Android Project введите следующие значения:

* Project name — LinearLayout Арр;
* Application name — LinearLayout Sample;
* Package name — сот. samples .linearlayout;

Далее в файле **res/layout/main.xml**создайте LinearLayout с тремя дочерними кнопками, как показано в листинге.

*Файл разметки main.xml для LinearLayout*

<?xml version="l.O" encoding="utf8"?>

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

android:id="@+id/LinearLayoutOl"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:orientation="horizontal">

<Button

android:id="@+id/buttonl"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="ButtonOl"/>

<Button

android:id="@+id/button2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Button02"/>

<Button

android:id="@+id/button3"

android:layout\_height="wrap\_content"

android:layout\_width="fill\_parent"

android:text="ButtonO3"/>

</LinearLayout>

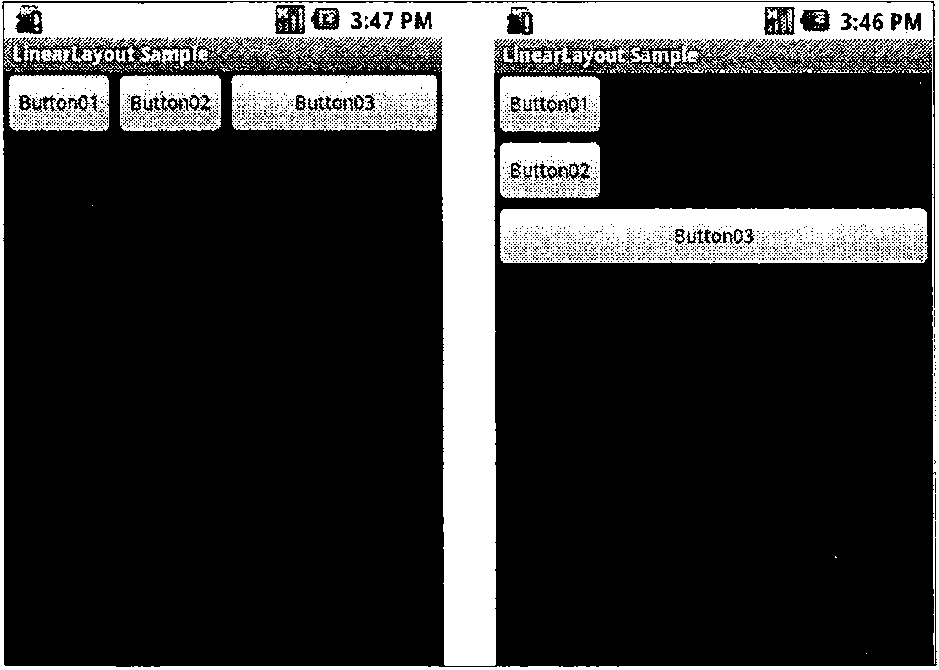
Обратите внимание, что у первых двух кнопок атрибуту **android: layoutwidth**присвоено значение wrapcontent, а у третьей кнопки — fillparent, т. е. последняя кнопка заполнит оставшееся свободное пространство в разметке.

В результате получится линейное горизонтальное размещение дочерних элементов. Если изменить в корневом элементе значение атрибута android: layout\_height:

android:orientation="vertical "

элементы в контейнере расположатся вертикально. Внешний вид экрана для разметки LinearLayout с горизонтальной и вертикальной ориентациями элементов показан на рисунке.

***Разметка LinearLayout*** также поддерживает атрибут android:layout\_weight, который назначает индивидуальный вес для дочернего элемента. Этот атрибут определяет "важность" представления и позволяет этому элементу расширяться, чтобы заполнить любое оставшееся пространство в родительском представлении. Заданный по умолчанию вес является нулевым.



*Разметка LinearLayout с горизонтальной и вертикальной ориентациями элементов*

Например, если есть три текстовых поля, и двум из них объявлен вес со зна ч^ием 1, в то время как другому не дается никакого веса (0), третье текстовое поле без веса не будет расширяться и займет область, определяемую размером текста, отображаемого этим полем. Другие два расширятся одинаково, чтобы заполнить остаток пространства, не занятого третьим полем. Если третьему полю присвоить вес 2 (вместо 0), это поле будет объявлено как "более важное", чем два других, так что третье поле получит 50% общего пространства, в то время как первые два получат по 25% общего пространства.

Для примера приложения с разметкой LinearLayout для окна и размещением на ней компонентов с разным "весом" создайте новый проект и в диалоговом окне New Android Project введите следующие значения:

* Project name — LinearLayoutWeightApp;
* Application name — TableLayout Sample;
* Package name — сот. samples. tablelayout;

Далее в файле res/layout/main.xml создайте LinearLayout с тремя дочерними виджетами EditText, как показано в листинге.

*Листинг. Файл разметки main.xml*

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

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

android:id="@+id/LinearLayout01"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:orientation="horizontal">

<EditText

android:id="@+id/EditText01"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="EditText01"

android:layout\_weight="0">

</EditText>

<EditText

android:id="@+id/EditText02"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="EditText02">

</EditText>

<EditText

android:id="@+id/EditText03"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="2"

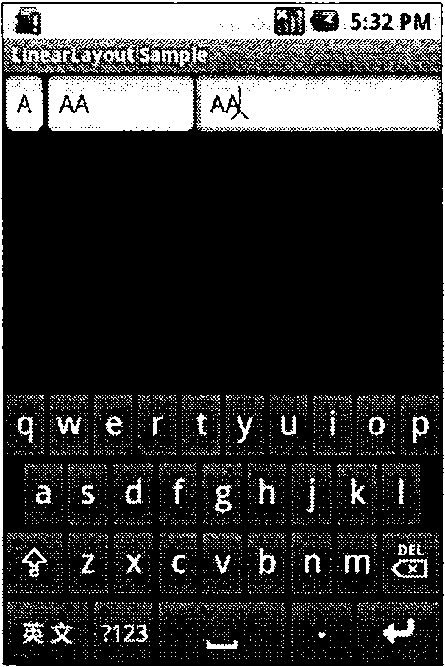
android:text="EditText03">

</EditText>

</LinearLayout>

Внешний вид экрана и влияние атрибута android: layoutweight Показаны на рисунке ниже.

Обратите внимание, как различные атрибуты XML определяют поведение представления. Попробуйте поэкспериментировать с различными значениями layout\_weight для дочерних элементов, чтобы увидеть, как будет распределяться доступное пространство для элементов.



*Отображение элементов с различными значениями android: layout\_weight*