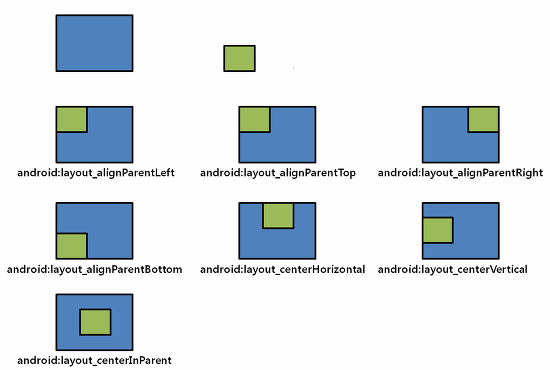
RelativeLayout

**ReiativeLayout** (относительная разметка) находится в разделе **Layouts** и позволяет дочерним компонентам определять свою позицию относительно родительского компонента или относительно соседних дочерних элементов (по идентификатору элемента). В **RelativeLayout** дочерние элементы расположены так, что если первый элемент расположен по центру экрана, другие элементы, выровненные относительно первого элемента, будут выровнены относительно центра экрана. При таком расположении, при объявлении разметки в XML-файле, элемент, на который будут ссылаться для позиционирования другие объекты представления, должен быть объявлен раньше, чем другие элементы, которые обращаются к нему по его идентификатору.

Важные атрибуты, которые связаны с родителем.

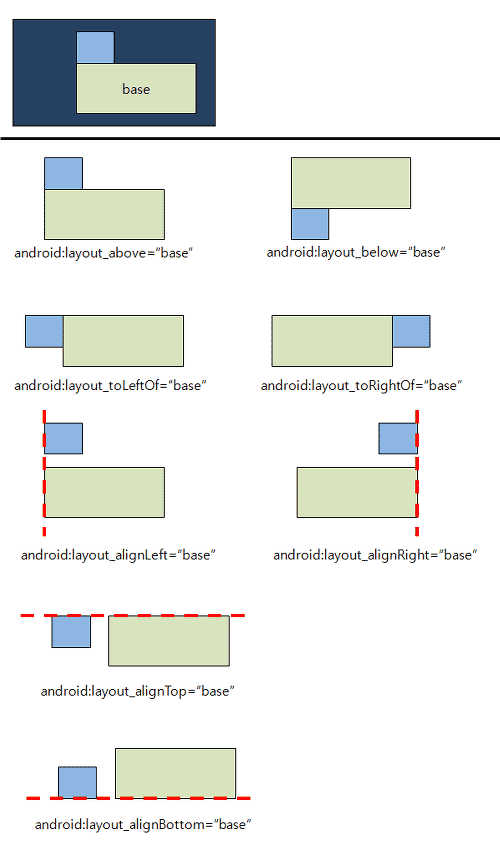
* **android:layout\_alignParentBottom** - выравнивание относительно нижнего края родителя
* **android:layout\_alignParentLeft** - выравнивание относительно левого края родителя
* **android:layout\_alignParentRight** - выравнивание относительно правого края родителя
* **android:layout\_alignParentTop** - выравнивание относительно верхнего края родителя
* **android:layout\_centerInParent** - выравнивание по центру родителя по вертикали и горизонтали
* **android:layout\_centerHorizontal** - выравнивание по центру родителя по горизонтали
* **android:layout\_centerVertical** - выравнивание по центру родителя по вертикали

Некоторые атрибуты можно использовать вместе, чтобы добиться нужного эффекта, например, разместить компонент в верхнем правом углу.



Компонент можно размещать не только относительно родителя, но и относительно других компонентов. Для этого все компоненты должны иметь свой идентификатор, по которому их можно будет отличать друг от друга. В этом случае вы можете задействовать другие атрибуты.

* **android:layout\_above** - размещается над указанным компонентом
* **android:layout\_below** - размещается под указанным компонентом
* **android:layout\_alignLeft** - выравнивается по левому краю указанного компонента
* **android:layout\_alignRight** - выравнивается по правому краю указанного компонента
* **android:layout\_alignTop** - выравнивается по верхнему краю указанного компонента
* **android:layout\_alignBottom** - выравнивается по нижнему краю указанного компонента
* **android:layout\_toLeftOf** - правый край компонента размещается слева от указанного компонента
* **android:layout\_toRightOf** - левый край компонент размещается справа от указанного компонента



Чтобы компоненты "не прилипали" друг к другу, используются атрибуты, добавляющие пространство между ними.

* **android:layout\_marginTop**
* **android:layout\_marginBottom**
* **android:layout\_marginLeft**
* **android:layout\_marginRight**

Рассмотрим простой пример. Допустим, мы хотим поместить на экран элементы **EditText** и **Button**. Кнопка должна находиться справа от текстового поля. Текстовое поле при этом должно быть выравнено слева относительно родительского элемента (компоновки) и слева относительно кнопки. В свою очередь кнопка должна быть выравнена справа относительно шаблона компоновки.

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" >

<EditText

android:id="@+id/editText"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_alignParentLeft="true"

android:layout\_toLeftOf="@+id/button"

android:hint="Введите имя кота..." >

</EditText>

<Button

android:id="@+id/button"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

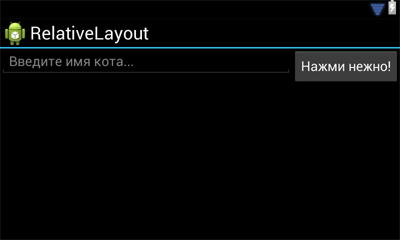
android:layout\_alignParentRight="true"

android:text="Нажми нежно!" >

</Button>

</RelativeLayout>

Во многих случаях грамотная относительная компоновка хорошо смотрится в альбомной и портретной ориентации.



Программное создание относительной разметки

Если вам понадобится динамически создавать относительную разметку в коде, то делается это следующим образом:

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

// setContentView(R.layout.activity\_main);

EditText editText = new EditText(this);

RelativeLayout.LayoutParams params = new RelativeLayout.LayoutParams(LayoutParams.MATCH\_PARENT,

LayoutParams.WRAP\_CONTENT);

params.addRule(RelativeLayout.ALIGN\_PARENT\_LEFT);

// use same id as defined when adding the button

params.addRule(RelativeLayout.LEFT\_OF, 1001);

editText.setLayoutParams(params);

editText.setHint("Введите имя кота....");

Button button = new Button(this);

RelativeLayout.LayoutParams params2 = new RelativeLayout.LayoutParams(LayoutParams.WRAP\_CONTENT,

LayoutParams.WRAP\_CONTENT);

params2.addRule(RelativeLayout.ALIGN\_PARENT\_RIGHT);

button.setLayoutParams(params2);

button.setText("Нажми нежно!");

// give the button an id that we know

button.setId(1001);

RelativeLayout layout = new RelativeLayout(this);

layout.setLayoutParams(new LayoutParams(LayoutParams.MATCH\_PARENT, LayoutParams.MATCH\_PARENT));

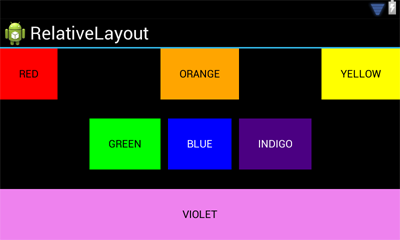
layout.addView(editText);

layout.addView(button);

setContentView(layout);

}

Создавать компоновку для относительной разметки чуть сложнее, чем для линейной компоновки. Рассмотрим ещё один пример. Предположим, нам нужен такой экран



Шаг 1: В XML-файле задаем относительную компоновку.

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" >

</RelativeLayout>

Шаг 2: Определяем дочерние элементы. В нашем случае мы имеем семь текстовых меток разного цвета. Каждому элементу присваиваем необходимые свойства: текст, цвет, размер символов и так далее.

Шаг 3: Определяем правила относительной разметки.  
Красная метка должна находиться в правом верхнем углу родительского элемента.  
Оранжевая метка должна располагаться по центру по горизонтали относительно родительского элемент.  
Желтая метка выравнивается по правой части родительского элемента.  
Зелёная метка выравнивается по центру (по вертикали) и выводится слева от синей метки.  
Синяя метка выравнивается по центру (по вертикали и горизонтали) относительно родительского элемента, т.е. точно по центру.  
Метка цвета индиго выравнивается по центру (по вертикали) и выводится справа от синей метки.  
Фиолетовая метка выводится в нижней части родительского элемента и занимает всю её ширину.

Код разметки:

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" >

<TextView

android:id="@+id/TextView01"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:background="#f00"

android:gravity="center"

android:padding="25dp"

android:text="RED"

android:textColor="#000" >

</TextView>

<TextView

android:id="@+id/TextView02"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_centerHorizontal="true"

android:background="#ffa500"

android:gravity="center"

android:padding="25dp"

android:text="ORANGE"

android:textColor="#000" >

</TextView>

<TextView

android:id="@+id/TextView03"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentRight="true"

android:background="#ffff00"

android:gravity="center"

android:padding="25dp"

android:text="YELLOW"

android:textColor="#000" >

</TextView>

<TextView

android:id="@+id/TextView04"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_centerVertical="true"

android:layout\_toLeftOf="@+id/TextView05"

android:background="#0f0"

android:gravity="center"

android:padding="25dp"

android:text="GREEN"

android:textColor="#000" >

</TextView>

<TextView

android:id="@+id/TextView05"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_centerInParent="true"

android:layout\_margin="10dp"

android:background="#00f"

android:gravity="center"

android:padding="25dp"

android:text="BLUE"

android:textColor="#fff" >

</TextView>

<TextView

android:id="@+id/TextView06"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_centerVertical="true"

android:layout\_toRightOf="@+id/TextView05"

android:background="#4b0082"

android:gravity="center"

android:padding="25dp"

android:text="INDIGO"

android:textColor="#fff" >

</TextView>

<TextView

android:id="@+id/TextView07"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:layout\_alignParentBottom="true"

android:background="#ee82ee"

android:gravity="center"

android:padding="25dp"

android:text="VIOLET"

android:textColor="#000" >

</TextView>

</RelativeLayout>

При разработке макета с относительной разметкой следите, чтобы элементы не налезали друг на друга (можете получить сообщение об ошибке). Старайтесь не усложнять макет и не забывайте присваивать уникальный идентификатор каждому элементу. Проверяйте макет в разных режимах ориентации.

Создаём неподвижную полоску

С помощью **RelativeLayout** можно создать неподвижную полоску внизу экрана, которая не будет прокручиваться вместе со списком. Для этого используется атрибут **android:layout\_alignParentBottom** и родственные ему атрибуты для верхней части. Вот простой пример со списком.

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical" >

<ListView

android:id="@+id/listView1"

android:layout\_width="match\_parent"

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

<LinearLayout

android:id="@+id/layout\_bottom\_bar"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:layout\_alignParentBottom="true"

android:background="#222222"

android:clickable="true"

android:gravity="center"

android:orientation="horizontal"

android:padding="5dip" >

<Button

android:id="@+id/button1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:background="#000000"

android:gravity="center"

android:padding="5dip"

android:text="Неподвижная часть"

android:textColor="#FFFFFF" />

</LinearLayout>

</RelativeLayout>

И простой код для создания прокручиваемого списка, чтобы увидеть эффект неподвижной полоски с кнопкой.

package ru.alexanderklimov.relativelayout;

import android.app.Activity;

import android.os.Bundle;

import android.widget.ArrayAdapter;

import android.widget.ListView;

public class RelativeLayoutActivity extends Activity {

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

ListView listView = (ListView) findViewById(R.id.listView1);

// определяем массив типа String

final String[] catnames = new String[] { "Рыжик", "Барсик", "Мурзик",

"Мурка", "Васька", "Томасина", "Кристина", "Пушок", "Дымка",

"Кузя", "Китти", "Масяня", "Симба" };

// используем адаптер данных

ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,

android.R.layout.simple\_list\_item\_1, catnames);

listView.setAdapter(adapter);

}

}

Запускаем проект и любуемся своей работой:

