LinearLayout — Layout, позволяющий разместить элементы вертикально или горизонтально друг за другом.

Давайте заменим код из прошлого урока на такой:

<?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:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal"

tools:context="com.test.lessons.MainActivity">

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="#ff0000" />

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="#00ff00" />

<View

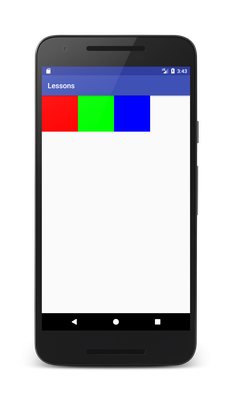
android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="#0000ff" />

</LinearLayout>

Запустив приложение, вы увидите, что View "выстроились" друг за другом по горизонтали, начиная с левого верхнего угла.

[](https://skillberg.com/media/uploads/2017/11/17/77f7325b859242d1b7257d097361525b.png)Горизонтальный LinearLayout

Как сделать, чтобы они выстроились по вертикали? Обратите внимание на атрибут android:orientation у LinearLayout. Он принимает следующие значения:

* horizontal — выстраивает дочерние View горизонтально
* vertical — выстраивает дочерние View вертикально

То есть, надо всего лишь заменить horizontal на vertical:

<?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:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

tools:context="com.test.lessons.MainActivity">

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="#ff0000" />

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="#00ff00" />

<View

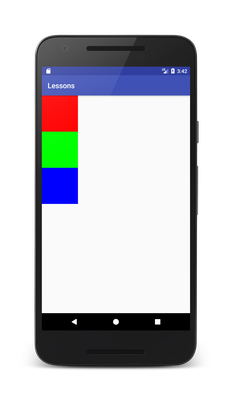
android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="#0000ff" />

</LinearLayout>

И получим вот такой результат:

[](https://skillberg.com/media/uploads/2017/11/17/8123de9fbbb943d2be4adee7b465a104.png)Вертикальный LinearLayout

Окей. А что еще можно сделать с LinearLayout? Мы можем задавать высоту и ширину в "процентах". То есть, например, я хочу, чтобы красный View занимал половину места по высоте, а синий и зеленый — по четверти.

Для этого используется атрибут layout\_weight:

<?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:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

tools:context="com.test.lessons.MainActivity">

<View

android:layout\_width="100dp"

android:layout\_height="0dp"

android:layout\_weight=".5"

android:background="#ff0000" />

<View

android:layout\_width="100dp"

android:layout\_height="0dp"

android:layout\_weight=".25"

android:background="#00ff00" />

<View

android:layout\_width="100dp"

android:layout\_height="0dp"

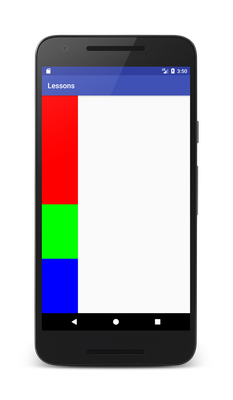
android:layout\_weight=".25"

android:background="#0000ff" />

</LinearLayout>

Получается, первый View имеет вес 0.5, а два остальных — по 0.25, в сумме — 1. Обратите внимание, что нужно обязательно задать высоту = 0, если у нас вертикальный LinearLayout. Если горизонтальный — то задаем ширину = 0. Почему? Потому что мы больше не контролируем высоту (или ширину в горизонтальном лэйауте), за нас это делает LinearLayout.

Посмотрим, что получилось:

[](https://skillberg.com/media/uploads/2017/11/17/3bd0e3bd944b41f9b6a3fb33710fd6d8.png)Использование layout\_weight в LinearLayout

Ура, сработало!

Кстати, сумма весов не обязательно должна быть равна 1. Просто так удобнее считать. А вообще, мы могли бы сделать что-то вроде такого:

<?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:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

tools:context="com.test.lessons.MainActivity">

<View

android:layout\_width="100dp"

android:layout\_height="0dp"

android:layout\_weight="30"

android:background="#ff0000" />

<View

android:layout\_width="100dp"

android:layout\_height="0dp"

android:layout\_weight="15"

android:background="#00ff00" />

<View

android:layout\_width="100dp"

android:layout\_height="0dp"

android:layout\_weight="15"

android:background="#0000ff" />

</LinearLayout>

И получить ровно такой же результат!

Еще мы можем сами задавать сумму весов. По-умолчанию сумма весов лэйаута считается как сумма всех значений layout\_weight, то есть подразумевается, что дочерние элементы суммарно занимают весь View в заданном направлении. Например, для предыдущего примера сумма весов равна 60, а для позапрошлого — 1.

Давайте вернемся к позапрошлому примеру, где сумма равна 1, и самостоятельно пропишем атрибут weightSum= 2:

<?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:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:weightSum="2"

tools:context="com.test.lessons.MainActivity">

<View

android:layout\_width="100dp"

android:layout\_height="0dp"

android:layout\_weight=".5"

android:background="#ff0000" />

<View

android:layout\_width="100dp"

android:layout\_height="0dp"

android:layout\_weight=".25"

android:background="#00ff00" />

<View

android:layout\_width="100dp"

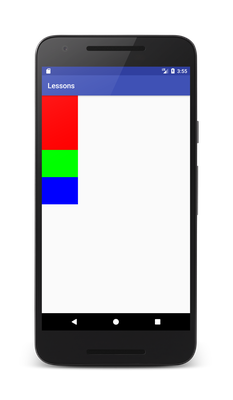
android:layout\_height="0dp"

android:layout\_weight=".25"

android:background="#0000ff" />

</LinearLayout>

Как думаете, что получится? А вот что:

[](https://skillberg.com/media/uploads/2017/11/17/a1914e1cec034f65abd72ddfda00a263.png)Задаём сумму весов вручную

Получается, суммарный вес всех дочерних элементов равен 1, но LinearLayout отталкивается от суммарного веса = 2, соответственно, он считает, что все дочерние элементы должны занимать ровно половину его высоты.

Наверное, это все, что можно рассказать про LinearLayout. В следующем уроке я расскажу вам про RelativeLayout — пожалуй, самый гибкий Layout.

* ПРИМЕРЫ КОДА
* [ИСХОДНЫЙ КОД УРОКА](https://github.com/Skillberg/LessonsAndroid/tree/lesson5)