**Всё о LinearLayout - 2**

Здравствуйте.  
Это вторая статья на тему разбора контейнера компоновки LinearLayout. Первая часть [здесь](http://devpad.ru/post/item/2/android-vse-o-linearlayout-1.html).

**Вес - layout\_weight**

layout\_weight очень важное свойство, которое позволяет назначить  элементу значимость размера относительно других элементов в контейнере.   
Предположим, что у нас, в горизонтальном направлении есть 3 кнопки и если мы их расположим друг за другом, то они займут только отведённое для них место и это может выглядеть далеко не пропорционально, а если мы назначим каждой кнопке атрибут android:layout\_weight="1", тогда все кнопки станут одинаковыми, но если мы хотим одинаковые по размеру левую и правую кнопки, а для средней лишь остаток места, тогда присвоим второй кнопке значение 0 для layout\_weight.  
  
Перейдём к примерам.

**листинг 1**

<?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"**>

<**LinearLayout** **android:layout\_width**=**"fill\_parent"**

**android:layout\_height**=**"fill\_parent"**

**android:orientation**=**"horizontal"**

**android:background**=**"#eee"**

**android:layout\_weight**=**"1"**>

<**Button** **android:layout\_width**=**"wrap\_content"**

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

**android:layout\_weight**=**"1"**

**android:text**=**"Вес кнопки 1"**/>

<**Button** **android:layout\_width**=**"wrap\_content"**

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

**android:layout\_weight**=**"1"**

**android:text**=**"Вес кнопки 1"**/>

</**LinearLayout**>

<**LinearLayout** **android:layout\_width**=**"fill\_parent"**

**android:layout\_height**=**"fill\_parent"**

**android:orientation**=**"horizontal"**

**android:background**=**"#ccc"**

**android:layout\_weight**=**"1"**>

<**Button** **android:layout\_width**=**"wrap\_content"**

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

**android:layout\_weight**=**"1"**

**android:text**=**"Вес кнопки 1"**/>

<**Button** **android:layout\_width**=**"wrap\_content"**

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

**android:layout\_weight**=**"1"**

**android:text**=**"Вес кнопки 1"**/>

<**Button** **android:layout\_width**=**"wrap\_content"**

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

**android:layout\_weight**=**"1"**

**android:text**=**"Вес кнопки 1"**/>

</**LinearLayout**>

<**LinearLayout** **android:layout\_width**=**"fill\_parent"**

**android:layout\_height**=**"fill\_parent"**

**android:orientation**=**"vertical"**

**android:background**=**"#aaa"**

**android:gravity**=**"center\_horizontal"**

**android:layout\_weight**=**"1"**>

<**Button** **android:layout\_width**=**"wrap\_content"**

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

**android:layout\_weight**=**"1"**

**android:text**=**"Вес кнопки 1"**/>

<**Button** **android:layout\_width**=**"wrap\_content"**

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

**android:layout\_weight**=**"1"**

**android:text**=**"Вес кнопки 1"**/>

<**Button** **android:layout\_width**=**"wrap\_content"**

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

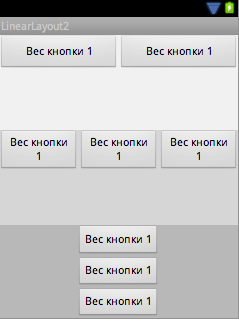
**android:layout\_weight**=**"1"**

**android:text**=**"Вес кнопки 1"**/>

</**LinearLayout**>

</**LinearLayout**>

Результат:



Итак, разберём пример.  
Для начала мы создаём 3 контейнера LinearLayout (внутри корневого). Чтобы визуально выделить их, зададим каждому свой фоновый цвет атрибутом android:background. Для того, чтобы эти контейнеры ровно распределились по нашему экрану, зададим каждому атрибуту android:layout\_weight значение равное 1 (по умолчанию установлено 0).   
Отлично! Далее в каждый контейнер мы добавили кнопки и также задали им значение атрибута android:layout\_weight равное 1. В первый контейнер с горизонтальной ориентацией мы добавили 2 кнопки, во второй 3 кнопки и в 3-ий с вертикальной ориентацией также 3 кнопки. Как видно из скриншота, они идеально распределили свободное пространство.

**Кстати**

*На Java установить атрибут веса можно в уже знакомом вам вложенном классе*LayoutParams*.  
Либо в конструкторе в 3-ем параметре*

LayoutParams param = **new** LinearLayout.LayoutParams(

LayoutParams.FILL\_PARENT,

LayoutParams.FILL\_PARENT,

1.0f

);

Либо изменив свойство weight

LinearLayout.LayoutParams params = **new** LinearLayout.LayoutParams(

LayoutParams.FILL\_PARENT, LayoutParams.FILL\_PARENT

);

params.weight = 1.0f;

Продолжим примеры.  
  
Что произойдёт, если мы будем изменять вес?  
Правильно! Элементы будут растягиваться.  
  
Рассмотрим следующий код:

**листинг 2 - вес по нарастающей**

<?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"**>

<**Button** **android:layout\_width**=**"fill\_parent"**

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

**android:layout\_weight**=**"0"**

**android:text**=**"Просто кнопка с весом 0"**/>

<**Button** **android:layout\_width**=**"fill\_parent"**

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

**android:layout\_weight**=**"1"**

**android:text**=**"Просто кнопка с весом 1"**/>

<**Button** **android:layout\_width**=**"fill\_parent"**

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

**android:layout\_weight**=**"2"**

**android:text**=**"Просто кнопка с весом 2"**/>

<**Button** **android:layout\_width**=**"fill\_parent"**

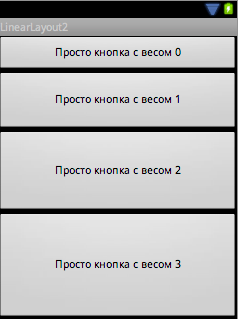
**android:layout\_height**=**"wrap\_content"**

**android:layout\_weight**=**"3"**

**android:text**=**"Просто кнопка с весом 3"**/>

</**LinearLayout**>

Сразу приложу результат кода:



Код достаточно простой.  
Есть 4 кнопки и каждой из них по нарастающей от 0 до 3-ёх устанавливаем вес.  
Те кнопки, что имеют больший вес, жадничают и поглощают как можно больше свободного пространства.

Рассмотрим ещё один пример.  
  
Нарисуем российский флаг во весь экран. Для этого нам понадобится 3 контейнера LinearLayout одинаковой высоты и разных цветов.  
Легко!

**листинг 3 - рисуем флаг**

<?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"**>

<**LinearLayout**

**android:layout\_weight**=**"1"**

**android:background**=**"#fff"**

**android:layout\_width**=**"fill\_parent"**

**android:layout\_height**=**"fill\_parent"**/>

<**LinearLayout**

**android:layout\_weight**=**"1"**

**android:background**=**"#00f"**

**android:layout\_width**=**"fill\_parent"**

**android:layout\_height**=**"fill\_parent"**/>

<**LinearLayout**

**android:layout\_weight**=**"1"**

**android:background**=**"#f00"**

**android:layout\_width**=**"fill\_parent"**

**android:layout\_height**=**"fill\_parent"**/>

</**LinearLayout**>

Вот, что у нас получилось:



Отлично!

Сделаем что-нибудь поинтереснее – закрепим линию с текстом сверху и снизу (footer и header), а внутри разместим длинный текст с возможностью его листать.

**листинг 4 – фиксированный header и footer**

<?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"**>

<**LinearLayout** **android:layout\_width**=**"fill\_parent"**

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

**android:orientation**=**"horizontal"**

**android:gravity**=**"center"**

**android:layout\_weight**=**"0"**

**android:background**=**"#ccc"**>

<**TextView** **android:layout\_width**=**"wrap\_content"**

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

**android:maxLines**=**"1"**

**android:padding**=**"10dp"**

**android:textColor**=**"#000"**

**android:text**=**"Fix header"**/>

</**LinearLayout**>

<**LinearLayout** **android:layout\_width**=**"fill\_parent"**

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

**android:orientation**=**"vertical"**

**android:layout\_weight**=**"1"**

**android:background**=**"#fff"**>

<**ScrollView** **android:layout\_width**=**"fill\_parent"**

**android:layout\_height**=**"fill\_parent"**>

<**TextView** **android:layout\_width**=**"wrap\_content"**

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

**android:textColor**=**"#000"**

**android:padding**=**"20dp"**

**android:text**=**"@string/long\_text"**/>

</**ScrollView**>

</**LinearLayout**>

<**LinearLayout** **android:layout\_width**=**"fill\_parent"**

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

**android:orientation**=**"horizontal"**

**android:gravity**=**"center"**

**android:layout\_weight**=**"0"**

**android:background**=**"#ccc"**>

<**TextView** **android:layout\_width**=**"wrap\_content"**

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

**android:maxLines**=**"1"**

**android:padding**=**"10dp"**

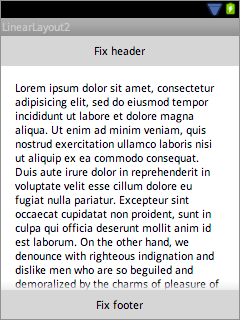
**android:textColor**=**"#000"**

**android:text**=**"Fix footer"**/>

</**LinearLayout**>

</**LinearLayout**>

**Результат:**



Рассмотрим, как это получилось:

1. В корневом контейнере создаём 3 контейнера LinearLayout и присваиваем среднему контейнеру вес равным 1.
2. Зададим первому и третьему контейнеру фоновый цвет в серый, а среднему в белый, для визуального разделения. Таким образом мы видим 2 тонкие полоски сверху и снизу, а в середине свободное место.
3. Добавим текст в первый и третий контейнер.
4. В среднем контейнере добавим длинный-длинный текст.
5. Чтобы текст можно было полностью пролистать, вложим его в виджет ScrollView.

У нас получилось некое подобие фиксированного [Action Bar](http://developer.android.com/design/get-started/ui-overview.html#app)'a и [Split Action Bar](http://developer.android.com/design/get-started/ui-overview.html#app)'а

Ну и последний пример, касающийся веса.

**Максимальный вес в группе - weightSum**

У контейнера LinearLayout можно установить максимальный вес с помощью атрибута weightSum.  
Если значение атрибута weightSum больше суммы всех атрибутов layout\_weight, тогда в контейнере будет добавлено свободное место.

Посмотрим на пример:

**листинг 5 – пример с weightSum**

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

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

**android:orientation**=**"vertical"**

**android:weightSum**=**"13"**

**android:gravity**=**"center\_horizontal"**

**android:layout\_width**=**"fill\_parent"**

**android:layout\_height**=**"fill\_parent"**>

<**Button** **android:layout\_width**=**"wrap\_content"**

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

**android:layout\_weight**=**"1"**

**android:text**=**"Weight 1"**/>

<**Button** **android:layout\_width**=**"wrap\_content"**

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

**android:layout\_weight**=**"2"**

**android:text**=**"Weight 2"**/>

<**Button** **android:layout\_width**=**"wrap\_content"**

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

**android:layout\_weight**=**"3"**

**android:text**=**"Weight 3"**/>

<**Button** **android:layout\_width**=**"wrap\_content"**

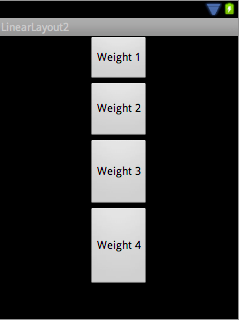
**android:layout\_height**=**"wrap\_content"**

**android:layout\_weight**=**"4"**

**android:text**=**"Weight 4"**/>

</**LinearLayout**>

Результат:



Если сложить все значения layout\_weight, то получим сумму 10. У нас максимальная сумма установлена в 13 и поэтому мы получаем свободное место после всех элементов.  
  
Чтобы установить свойство weightSum на JAVA, нужно воспользоваться методом setWeightSum (float weightSum)

linearLayout.setWeightSum(13.0f)

В следующей статье, я расскажу про тяготение (выравнивание) – атрибуты layout\_gravity и gravity, и про выравнивание по базовой линии – атрибутыbaselineAligned и baselineAlignedChildIndex.  
Исходники с примерами будут выложены на гитхаб в следующей статье.