**Контейнеры компоновки**

Контейнеры используются для установки положения дочерних представлений в создаваемом пользовательском интерфейсе. Вы можете последовательно вкладывать макеты и, следовательно, создавать произвольно сложные пользовательские интерфейсы, используя комбинацию макетов.

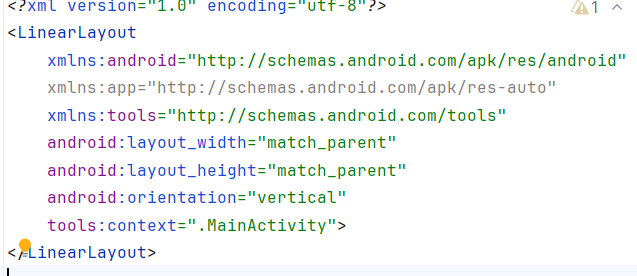
Основные типы контейнеров:

* LinearLayout - Используемый для предоставления дочерних элементов один за другим в определенном направлении, по горизонтали или по вертикали, в зависимости от свойства orientation.
* RelativeLayout - используется для указания положения дочерних элементов относительно друг друга, например (A справа от B) или относительно родительского элемента (фиксация вверху родительского элемента).
* ConstraintLayout - используется для указания положения для каждого дочернего элемента относительно других присутствующих элементов. ConstraintLayout похож на RelativeLayout, но имеет больше возможностей.
* FrameLayout - используется для указания положения содержащихся в нем элементов друг над другом.
* TableLayout - используется для отображения дочерних элементов в строках и столбцах в виде таблицы.
* GridView - Используется для отображения прокручиваемого списка элементов в виде сетки строк и столбцов.
* ListView - Используется для отображения прокручиваемых списков элементов в одном столбце.
* WebView - это браузер, который используется для отображения веб-страниц.

Рассмотрим подробнее некоторые из них

**LinearLayout**

LinearLayout — это самый простой макет в студии Android, который последовательно выравнивает всех дочерних элементов по горизонтали или по вертикали путем указания атрибута orientation . Если вы примените orientation="vertical" , то элементы будут располагаться один за другим по вертикали, а если вы примените orientation="horizontal" , то элементы будут располагаться один за другим по горизонтали.



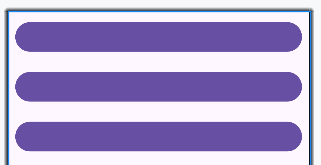
Атрибуты:

* layout\_weight - Определяется в дочерних элементах, чтобы указать, как LinearLayout разделяет пространство между содержащимися в нем элементами
* weightSum - Определяет общую сумму веса
* orientation - Определяет как должны быть расположены элементы в макете. Может быть горизонтальным или вертикальным.
* gravity - Определяет, как объект должен располагать свое содержимое по осям X и Y.
* layout\_gravity - Устанавливает гравитацию элемента относительно его родителя.

Примеры:

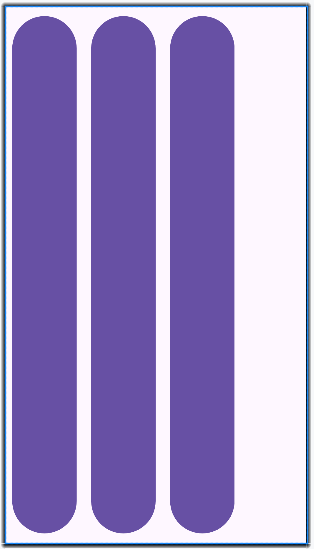
Расположение по вертикали:





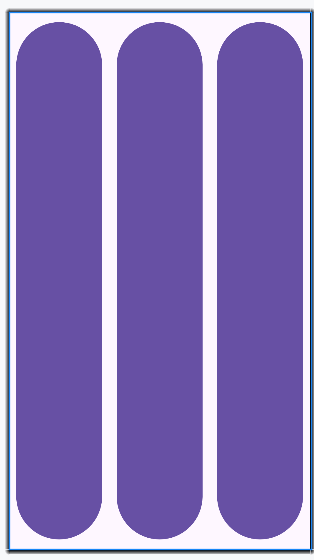
Расположение по диагонали:





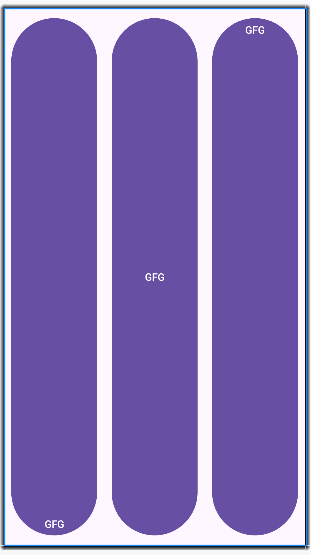
Использование веса:





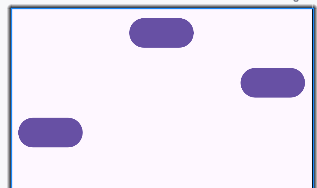
Использование gravity:





Использование layout\_gravity:





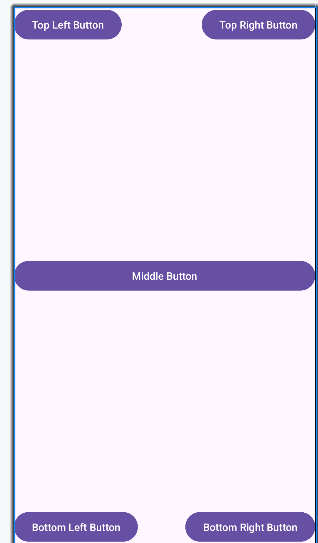
**RelativeLayout**

RelativeLayout используется для упорядочения дочерних элементов, относительно друг друга. Например, создадим приложение с LinearLayout, состоящим из 5 элементов. Даже если мы укажем свойства веса и гравитации, они не будут расположены относительно друг друга, они будут расположены подряд как они написаны в коде xml. Чтобы их упорядочить так как необходимо, нам нужны дополнительные свойства.

Ключевым свойством является семейство layout\_alignParent… .Он определяет относительно какой стороны родительского контейнера необходимо расположить элемент, это свойство определяется в названии атрибута (layout\_alignParentTop, layout\_alignParentBottom, layout\_alignParentRight, layout\_alignParentLeft)

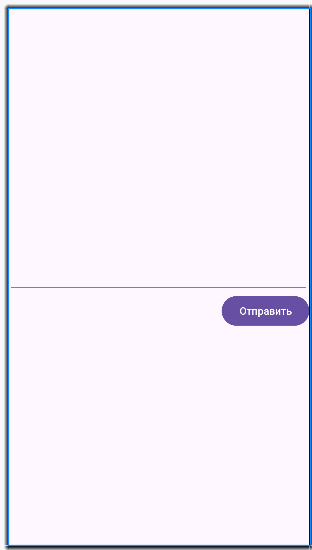






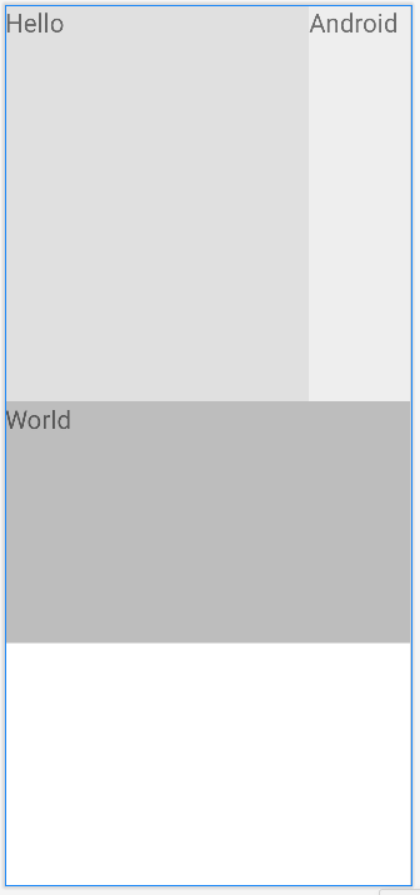
Еще одним важным свойством является семейство layout\_align… .В его значение устанавливается идентификатор относительно какого элемента необходимо расположить элемент,название атрибута определет с какой стороны расположить элемент (layout\_alignBottom, layout\_alignLeft, layout\_alignRight, layout\_alignTop)





**Задание для самостоятельной работы:**

1. Разработайте приложение с интерфейсом соответствующим следующему макету (используя LinearLayout)



1. Разработайте приложение с интерфейсом соответствующим следующему макету (используя RelativeLayout)

