**[Layouts (разметка)](http://ru-code-android.livejournal.com/4327.html" \t "_self)**

* May. 23rd, 2012 at 12:15 AM

  
[http://l-stat.livejournal.net/img/userinfo.gif?v=17080?v=138.8](http://coderka.livejournal.com/profile)[**coderka**](http://coderka.livejournal.com/)

Любая Activity (деятельность, окно приложения) состоит из Views и ViewGroups. View — виджет, который рисуется на экране, например, кнопка, текстовое поле, список. Все такие виджеты наследуются от класса android.view.View.   
ViewGroup — разметка, позволяющая расположить один или несколько View. Разметки наследуются от класса android.view.ViewGroup, который в свою очередь наследуется от android.view.View. Это значит что дочерними элементами разметки могут быть не только View, но и сами разметки  
  
**Примеры ViewGroups:**  
- FrameLayout  
- LinearLayout  
- TableLayout  
- RelativeLayout  
- ScrollView  
  
Рассмотрим их более подробно. 

**FrameLayout**

FrameLayout — разметка для отображения одного элемента. Дочерние View или ViewGroup в FrameLayout выравниваются по верхнему левому углу. Разметка может содержать несколько элементов, но тогда они будут перекрывать друг друга. Например:

|  |  |
| --- | --- |
| 1. <FrameLayout 2. android:layout\_width="fill\_parent" 3. android:layout\_height="fill\_parent"> 5. <Button 6. android:layout\_width="wrap\_content" 7. android:layout\_height="wrap\_content" 8. android:text="@string/Button" 9. /> 10. </FrameLayout> | http://pics.livejournal.com/coderka/pic/0004dr63 |

|  |  |
| --- | --- |
| 1. <FrameLayout 2. android:layout\_width="fill\_parent" 3. android:layout\_height="fill\_parent"> 5. <ImageView 6. android:layout\_width="wrap\_content" 7. android:layout\_height="wrap\_content" 8. android:src="@drawable/image" 9. /> 10. <Button 11. android:layout\_width="wrap\_content" 12. android:layout\_height="wrap\_content" 13. android:text="@string/Button" 14. /> 15. </FrameLayout> | http://pics.livejournal.com/coderka/pic/0004e4tq |

FrameLayout удобно использовать для нескольких элементов, которые скрываются и показываются программно, таким образом что в каждый момент времени видимым оказывается только один из них. Для управления видимостью элемента используется атрибут *android:visibility* в xml или метод *setVisibility (int visibility)*, где visibility принимает одно из трех значений:  
VISIBLE — элемент видим  
INVISIBLE — элемент невидим, но все еще занимает место в разметке  
GONE — элемент невидим и не занимает место в разметке

**LinearLayout**

Разметка для отображения одного или нескольких элементов в одну линию, горизонтально или вертикально. Для выбора ориентации используется атрибут android:orientation с двумя возможными значениями «horizontal» и «vertical»

|  |  |
| --- | --- |
| 1. <LinearLayout 2. android:layout\_width="fill\_parent" 3. android:layout\_height="fill\_parent" 4. android:orientation="vertical" 5. > 6. <Button 7. android:layout\_width="wrap\_content" 8. android:layout\_height="wrap\_content" 9. android:text="@string/Button" 10. /> 11. <Button 12. android:layout\_width="wrap\_content" 13. android:layout\_height="wrap\_content" 14. android:text="@string/Button" 15. /> 16. <Button 17. android:layout\_width="wrap\_content" 18. android:layout\_height="wrap\_content" 19. android:text="@string/Button" 20. /> 21. </LinearLayout> | http://pics.livejournal.com/coderka/pic/0004f7xh |

|  |  |
| --- | --- |
| 1. <LinearLayout 2. android:layout\_width="fill\_parent" 3. android:layout\_height="fill\_parent" 4. android:orientation="horizontal" 5. > 6. <Button 7. android:layout\_width="wrap\_content" 8. android:layout\_height="wrap\_content" 9. android:text="@string/Button" 10. /> 11. <Button 12. android:layout\_width="wrap\_content" 13. android:layout\_height="wrap\_content" 14. android:text="@string/Button" 15. /> 16. <Button 17. android:layout\_width="wrap\_content" 18. android:layout\_height="wrap\_content" 19. android:text="@string/Button" 20. /> 21. </LinearLayout> | http://pics.livejournal.com/coderka/pic/0004g6k6 |

**TableLayout**

Разметка для расположения элементов в виде таблицы. Ряды задаются в xml с помощью тега *TableRow*, а ячейки в каждом ряду создаются автоматически для каждого элемента. Количество колонок в таблице будет равно максимальному количеству элементов в рядах. То есть если в одном ряду 3 элемента, а во втором 1, то колонок в TableLayout будет 3. Ширина колонки определяется по самому широкому элементу в ней. Ячейки таблицы можно оставлять пустыми, или объединять. Атрибут *android:layout\_column* — задает в какой столбец поместить данный элемент (нумерация столбцов начинается с 0). Атрибут *android:layout\_span* — позволяет объединить указанное количество столбцов (ряды объединить нельзя)

|  |  |
| --- | --- |
| 1. <TableLayout 2. android:layout\_width="fill\_parent" 3. android:layout\_height="fill\_parent" 4. > 5. <TableRow> 6. <ImageView 7. android:layout\_width="wrap\_content" 8. android:layout\_height="wrap\_content" 9. android:src="@drawable/image" 10. android:layout\_column="1" 11. /> 12. </TableRow> 14. <TableRow> 15. <Button 16. android:layout\_width="wrap\_content" 17. android:layout\_height="wrap\_content" 18. android:text="@string/Button" 19. /> 20. <Button 21. android:layout\_width="wrap\_content" 22. android:layout\_height="wrap\_content" 23. android:text="@string/Button" 24. /> 25. </TableRow> 27. <TableRow> 28. <Button 29. android:layout\_width="wrap\_content" 30. android:layout\_height="wrap\_content" 31. android:text="@string/Button" 32. android:layout\_span="2" 33. /> 34. </TableRow> 35. </TableLayout> | http://pics.livejournal.com/coderka/pic/0004hf1x |

**RelativeLayout**

Разметка для расположения элементов относительно родителя или друг друга. Элементы начинают располагаться в указанном порядке, поэтому необходимо чтобы элемент был описан до того, как другой элемент будет на него ссылаться.

|  |  |
| --- | --- |
| 1. <RelativeLayout 2. android:layout\_width="fill\_parent" 3. android:layout\_height="fill\_parent"> 5. <ImageView 6. android:id="@+id/image" 7. android:layout\_width="wrap\_content" 8. android:layout\_height="wrap\_content" 9. android:src="@drawable/image" 10. android:layout\_centerHorizontal="true" 11. /> 12. <Button 13. android:layout\_width="wrap\_content" 14. android:layout\_height="wrap\_content" 15. android:layout\_toLeftOf="@id/image" 16. android:text="@string/Left" 17. /> 18. <Button 19. android:layout\_width="wrap\_content" 20. android:layout\_height="wrap\_content" 21. android:layout\_toRightOf="@id/image" 22. android:text="@string/Right" 23. /> 24. </RelativeLayout> | http://pics.livejournal.com/coderka/pic/0004ksce |

Возможные атрибуты элементов:  
**Расположение относительно родительского элемента.** Значением атрибута является «true»  
*android:layout\_alignParentTop* — верхняя граница элемента располагается на верхней границе RelativeLayout  
*android:layout\_alignParentLeft* — левая граница элемента располагается на левой границе RelativeLayout  
*android:layout\_alignParentRight* — правая граница элемента располагается на правой границе RelativeLayout  
*android:layout\_alignParentBottom* — нижняя граница элемента располагается на нижней границе RelativeLayout  
*android:layout\_centerInParent* — элемент располагается в центре родительского по горизонтали и вертикали  
*android:layout\_centerHorizontal* — элемент располагается в центре родительского по горизонтали  
*android:layout\_centerVertical* — элемент располагается в центре родительского по вертикали  
  
**Расположение относительно других элементов**, значением атрибута является id другого элемента.  
*android:layout\_above* — элемент располагается сверху от другого элемента  
*android:layout\_toLeftOf* — элемент располагается слева от другого элемента  
*android:layout\_toRightOf* — элемент располагается справа от другого элемента.  
*android:layout\_below* — элемент располагается снизу от другого элемента.  
*android:layout\_alignBaseline* — базовая линия элемента выравнивается по базовой линии другого элемента  
*android:layout\_alignTop* — верхняя граница элемента выравнивается по верхней границе другого элемента  
*android:layout\_alignLeft* — левая граница элемента выравнивается по левой границе другого элемента  
*android:layout\_alignRight* — правая граница элемента выравнивается по правой границе другого элемента  
*android:layout\_alignBottom* — нижняя граница элемента выравнивается по нижней границе другого элемента

**ScrollView**

ScrollView является наследником класса FrameLayout. Отличием является то, что он позволяет прокручивать элементы, если они занимают больше места чем физический размер экрана. В ScrollView также может содержаться только один элемент (View или ViewGroup), чаще всего используется LinearLayout, в которого вложено несколько элементов. Scrollview поддерживает только вертикальную прокрутку.

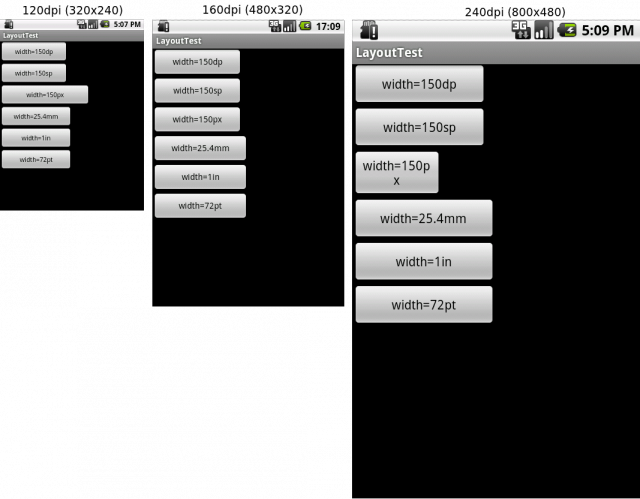
|  |  |
| --- | --- |
| 1. <ScrollView 2. android:layout\_width="fill\_parent" 3. android:layout\_height="fill\_parent"> 4. <LinearLayout 5. android:layout\_width="fill\_parent" 6. android:layout\_height="wrap\_content" 7. android:orientation="vertical"> 8. <!-- Много элементов --> 9. </LinearLayout> 10. </ScrollView> | http://pics.livejournal.com/coderka/pic/0004pa25 |

Мы рассмотрели основные виды разметок в android. Комбинируя и вкладывая их друг в друга можно создать интерфейс любой сложности.

**Атрибуты элементов**

Во всех View которые вкладываются в ViewGroup есть атрибуты позволяющие управлять их размерами или расположением. Основные из них:

**android:layout\_width и android:layout\_height**

Ширина и высота элемента. Атрибут может принимать следующие значения:  
**fill\_parent** — элемент имеет размер такой же как и родительский элемент  
**wrap\_content** — элемент имеет размер по своему содержимому  
**точное значение размера**. Размер может указываться в следующих единицах:  
*dp* - (Density-independent Pixels) - абстрактная единица, основанная на плотности экрана. На устройствах с экраном 160dpi: 1dp=1px. Это соотношение будет меняться на экранах с другой плотностью, причем необязательно в прямой пропорции. Таким образом обеспечивается согласованность размеров элементов интерфейса на различных устройствах.  
  
*sp* - (Scale-independent Pixels) — единица измерения похожая на dp, однако она также масштабируется учитывая пользовательские настройки шрифта.   
  
*px* — пиксели. Не рекомендуется использовать эту единицу измерения, так как представление элемента может различаться в зависимости от устройства. Так если мы задаем ширину элемента 100px, то на устройствах с экраном 320x240 это будет почти полэкрана, а на устройствах 800x480 - меньше четверти экрана.  
  
*mm* - миллиметры  
  
*in* - дюймы  
  
*pt* — (points) 1/72 дюйма  
  
Последние три единицы измерения не зависят от разрешения и диагонали экрана.  
1in = 25.4mm = 72pt  
  
На скриншоте показано как выглядят кнопки на разных размерах экрана  
[](http://pics.livejournal.com/coderka/pic/0004r48b/)

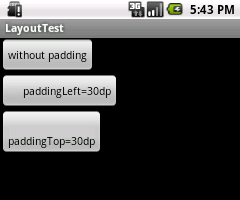
**android:layout\_gravity**

Атрибут определяющий как данный View позиционируется внутри родительского элемента. Основные значения: top, bottom, left, right, center. Значение у элемента может быть одно или несколько (тогда они разделяются «|»).   
Важно не путать android:layout\_gravity и android:gravity. android:gravity — определяет позиционирование внутри самого элемента, а android:layout\_gravity — позиционирование элемента внутри родительского.

**android:layout\_margin**

А также android:layout\_marginLeft, android:layout\_marginRight, android:layout\_marginTop, android:layout\_marginBottom. Атрибут определяет отступ элемента от соседних элементов (или от границы родительского)

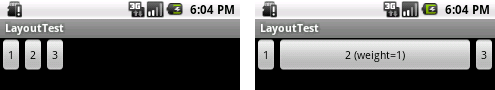
|  |  |
| --- | --- |
| 1. <LinearLayout 2. android:layout\_width="fill\_parent" 3. android:layout\_height="fill\_parent" 4. android:orientation="vertical"> 6. <Button 7. android:layout\_width="wrap\_content" 8. android:layout\_height="wrap\_content" 9. android:text="without margin" 10. /> 12. <Button 13. android:layout\_width="wrap\_content" 14. android:layout\_height="wrap\_content" 15. android:text="marginLeft=30dp" 16. android:layout\_marginLeft="30dp" 17. /> 19. <Button 20. android:layout\_width="wrap\_content" 21. android:layout\_height="wrap\_content" 22. android:text="marginTop=30dp" 23. android:layout\_marginTop="30dp" 24. /> 25. </LinearLayout> | http://pics.livejournal.com/coderka/pic/0004s701 |

Для отступа внутри элемента используются атрибуты   
android:padding, android:paddingLeft, android:paddingRight, android:paddingTop, android:paddingBottom  


**android:weight**

Атрибут определяет соотношение в котором будет распределятся пространство для элементов разметки.

|  |  |
| --- | --- |
| 1. <LinearLayout 2. android:layout\_width="fill\_parent" 3. android:layout\_height="fill\_parent" 4. android:orientation="horizontal"> 6. <Button 7. android:layout\_width="wrap\_content" 8. android:layout\_height="wrap\_content" 9. android:text="weight=1" 10. android:layout\_weight="1" 11. /> 13. <Button 14. android:layout\_width="wrap\_content" 15. android:layout\_height="wrap\_content" 16. android:text="weight=2" 17. android:layout\_weight="2" 18. /> 19. </LinearLayout> | http://pics.livejournal.com/coderka/pic/0004w8qy |

У первой кнопки weight=1, у второй weight=2, таким образом ширина первой 1/3 экрана, а второй 2/3 экрана. На скриншоте это соотношение может быть неочевидным, так как при отрисовки элементов учитывается их ширина (в нашем случае wrap\_content). Если поставить ширину обеим кнопкам android:layout\_width="0dp", то вторая кнопка будет ровно в два раза длиннее первой. Если мы хотим чтобы длина кнопок была одинаковой, необходимо поставить им одинаковый вес (причем неважно какой).  
  
Если атрибут layout\_weight поставить только у одного элемента, то он займет максимум свободного пространства. Так на первом скриншоте у трех кнопок android:layout\_width="wrap\_content" и не задан weight. На втором скриншоте у второй кнопки задан android:layout\_weight="1"  


Облако тегов:

* [framelayout](http://ru-code-android.livejournal.com/tag/framelayout),
* [gravity](http://ru-code-android.livejournal.com/tag/gravity),
* [layout](http://ru-code-android.livejournal.com/tag/layout),
* [linearlayout](http://ru-code-android.livejournal.com/tag/linearlayout),
* [relativelayout](http://ru-code-android.livejournal.com/tag/relativelayout),
* [scrollview](http://ru-code-android.livejournal.com/tag/scrollview),
* [tablelayout](http://ru-code-android.livejournal.com/tag/tablelayout),
* [weight](http://ru-code-android.livejournal.com/tag/weight)