LinearLayout

Последнее обновление:

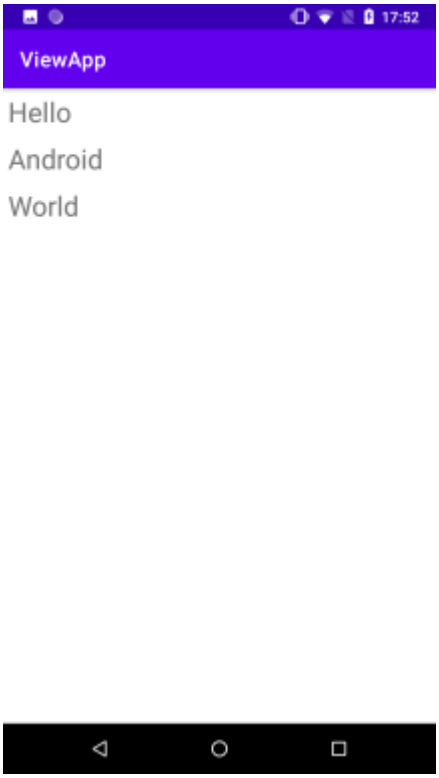
Контейнер **LinearLayout** представляет простейший контейнер - объект `ViewGroup`, который упорядочивает все дочерние элементы в одном направлении: по горизонтали или по вертикали. Все элемены расположены один за другим. Направление разметки указывается с помощью атрибута **android:orientation**.

Если, например, ориентация разметки вертикальная (`android:orientation="vertical"`), то все элементы располагаются в столбик - по одному элементу на каждой строке. Если ориентация горизонтальная (`android:orientation="horizontal"`), то элементы располагаются в одну строку. Например, расположим элементы в горизонтальный ряд:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="5dp"
        android:text="Hello"
```

```
        android:textSize="26sp" />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="5dp"
        android:text="Android"
        android:textSize="26sp" />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="5dp"
        android:text="World"
        android:textSize="26sp" />
</LinearLayout>
```

Если бы мы указали для LinearLayout атрибут **android:orientation="vertical"**, то элементы размещались бы по вертикали:



Вес элемента

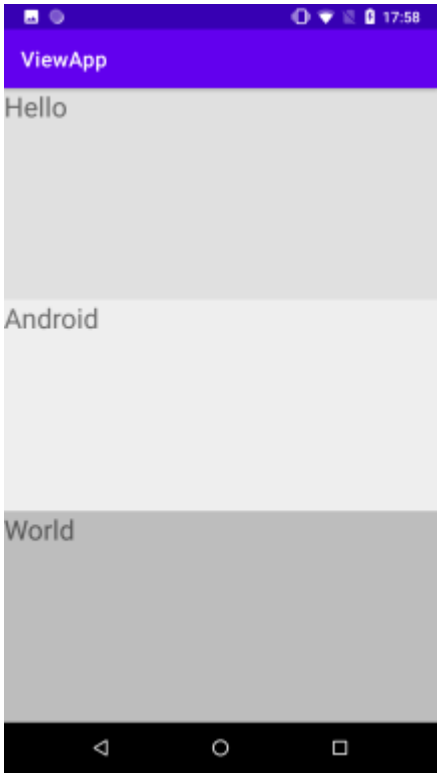
LinearLayout поддерживает такое свойство, как вес элемента, которое передается атрибутом **android:layout_weight**. Это свойство принимает значение, указывающее, какую часть оставшегося свободного места контейнера по отношению к другим объектам займет данный элемент. Например, если один элемент у нас будет иметь для свойства `android:layout_weight` значение 2, а другой - значение 1, то в сумме они дадут 3, поэтому первый элемент будет занимать 2/3 оставшегося пространства, а второй - 1/3.

Если все элементы имеют значение `android:layout_weight="1"`, то все эти элементы будут равномерно распределены по всей площади контейнера:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <TextView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:text="Hello"
        android:background="#e0e0e0"
        android:layout_weight="1"
        android:textSize="26sp" />
    <TextView
        android:layout_width="match_parent"
        android:layout_height="0dp"
```

```
        android:background="#eeeeee"
        android:text="Android"
        android:layout_weight="1"
        android:textSize="26sp" />
    <TextView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:text="World"
        android:background="#bdbdbd"
        android:layout_weight="1"
        android:textSize="26sp" />
</LinearLayout>
```

В данном случае LinearLayout имеет вертикальную ориентацию, поэтому все элементы будут располагаться сверху вниз. Все три элемента имеют значение `android:layout_weight="1"`, поэтому сумма весов всех элементов будет равна 3, а каждый элемент получит по трети пространства в LinearLayout:



При этом так как у нас вертикальный стек, то нам надо также установить для свойства `layout_height` значение **0dp**. Если бы LinearLayout имел горизонтальную ориентацию, то для свойства `layout_width` надо было бы установить значение **0dp**.

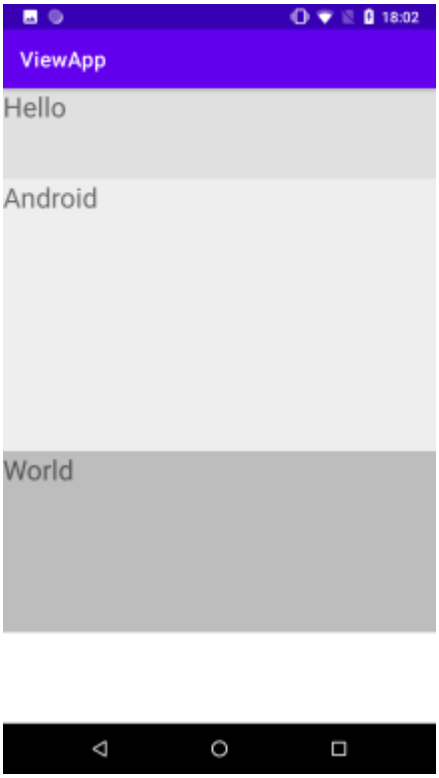
Еще один атрибут **android:weightSum** позволяет указать сумму весов всех элементов. Например:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="7">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:text="Hello"
        android:background="#e0e0e0"
        android:layout_weight="1"
        android:textSize="26sp" />
    <TextView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:background="#eeeeee"
        android:text="Android"
        android:layout_weight="3"
        android:textSize="26sp" />
    <TextView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:text="World"
        android:background="#bdbdbd"
        android:layout_weight="2"
        android:textSize="26sp" />
</LinearLayout>
```

LinearLayout здесь задает сумму весов равную 7. То есть все пространство по вертикали (так как вертикальная ориентация) условно делится на семь равных частей.

Первый TextView имеет вес 1, то есть из этих семи частей занимает только одну. Второй TextView имеет вес 3, то есть занимает три части из семи. И третий имеет вес 2. Итоговая сумма составляет 6. Но так как LinearLayout задает вес 7, то одна часть будет свободна от всех элементов.

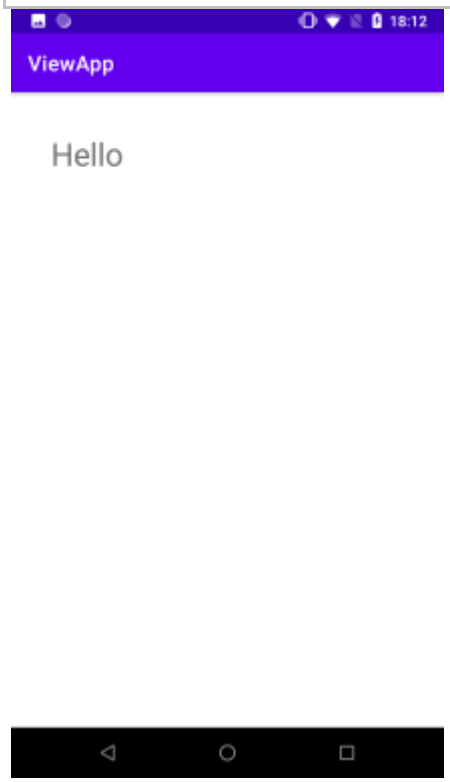


Программное создание LinearLayout

Создание LinearLayout в коде java:

```
1  package com.example.viewapp;
2
3  import androidx.appcompat.app.AppCompatActivity;
4  import android.os.Bundle;
5  import android.widget.LinearLayout;
6  import android.widget.TextView;
7
8  public class MainActivity extends AppCompatActivity {
9
10     @Override
11     protected void onCreate(Bundle savedInstanceState) {
12         super.onCreate(savedInstanceState);
13         //setContentView(R.layout.activity_main);
14         LinearLayout linearLayout = new LinearLayout(this);
15         // горизонтальная ориентация
16         linearLayout.setOrientation(LinearLayout.HORIZONTAL);
17
18         TextView textView = new TextView(this);
19         textView.setText("Hello");
20         textView.setTextSize(30);
21         // создаем параметры позиционирования для элемента
22         LinearLayout.LayoutParams layoutParams = new LinearLayout.LayoutPara
23                 (LinearLayout.LayoutParams.WRAP_CONTENT, LinearLayout.Layou
24         // устанавливаем отступы
25         layoutParams.setMargins(100, 100, 0, 0);
```

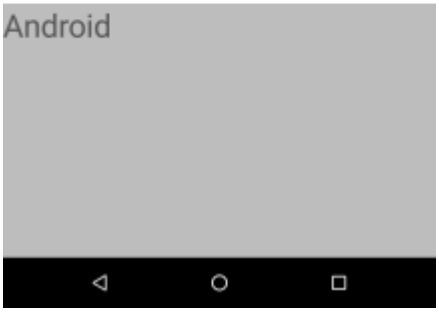
```
        textView.setLayoutParams(layoutParams);
        // добавляем элемент в LinearLayout
        linearLayout.addView(textView);

        setContentView(linearLayout);
    }
}
```



Дополнительная версия конструктора **LinearLayout.LayoutParams()** в качестве третьего параметра позволяет указать вес элемента:

```
package com.example.viewapp;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.LinearLayout;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        LinearLayout linearLayout = new LinearLayout(this);
        linearLayout.setOrientation(LinearLayout.VERTICAL);

        // первое текстовое поле
        TextView textView1 = new TextView(this);
```

```
        textView1.setText("Hello");
        textView1.setTextSize(30);
        // textView1 имеет вес 3
        linearLayout.addView(textView1, new LinearLayout.LayoutParams
                (LinearLayout.LayoutParams.MATCH_PARENT, 0, 3));

        // второе текстовое поле
        TextView textView2 = new TextView(this);
        textView2.setText("Android");
        textView2.setBackgroundColor(0xFFBDBDBD);
        textView2.setTextSize(30);
        // textView2 имеет вес 2
        linearLayout.addView(textView2, new LinearLayout.LayoutParams
                (LinearLayout.LayoutParams.MATCH_PARENT, 0, 2));

        setContentView(linearLayout);
    }
}
```



Layout_gravity

Атрибут **layout_gravity** позволяет устанавливать позиционирование относительно LinearLayout. Он принимает следуюшие значения:

- `top`: выравнивает элемент по верхней границе контейнера
- `bottom`: выравнивает элемент по нижней границе контейнера
- `left`: выравнивает элемент по левой границе контейнера

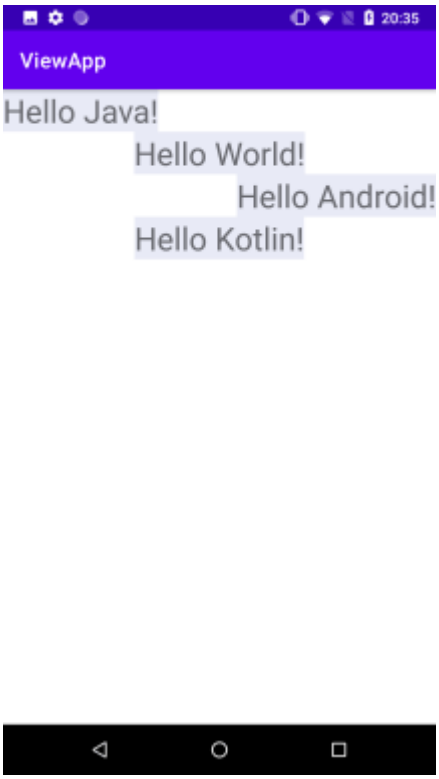
- `right`: выравнивает элемент по правой границе контейнера
- `center_vertical`: выравнивает элемент по центру по вертикали
- `center_horizontal`: выравнивает элемент по центру по горизонтали
- `center`: элемент позиционируется в центре
- `fill_vertical`: элемент растягивается по вертикали
- `fill_horizontal`: элемент растягивается по горизонтали
- `fill`: элемент заполняет все пространство контейнера
- `clip_vertical`: обрезает верхнюю и нижнюю границу элемента
- `clip_horizontal`: обрезает правую и левую границу элемента
- `start`: элемент позиционируется в начале (в верхнем левом углу) контейнера
- end: элемент позиционируется в конце контейнера (в верхнем правом углу)

Например:

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <TextView
        android:layout_gravity="left"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:text="Hello Java!"
        android:background="#e8eaf6"/>
    <TextView
        android:layout_gravity="center"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:text="Hello World!"
        android:background="#e8eaf6"/>
    <TextView
        android:layout_gravity="right"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:text="Hello Android!"
```

```
        android:background="#e8eaf6"/>
    <TextView
        android:layout_gravity="center"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:text="Hello Kotlin!"
        android:background="#e8eaf6"/>
</LinearLayout>
```

В данном случае первый элемент TextView будет позиционироваться по левой стороне контейнера (`android:layout_gravity="left"`), второй TextView по центру (`android:layout_gravity="center"`), третий - по правой стороне (`android:layout_gravity="right"`) и четвертый - по центру (`android:layout_gravity="center"`)



Стоит учитывать ориентацию контейнера. Например, при вертикальной ориентации все элементы будут представлять вертикальный стек, идущий сверху вниз. Поэтому значения, которые относятся к позиционированию элемента по вертикали (например, top или bottom) никак не будут влиять на элемент. Также при горизонтальной ориентации LinearLayout не окажут никакого влияния значения, которые позиционируют элемент по горизонтали, например, left и right.

Для установки программно параметра **layout_gravity** надо задать поле **gravity** у объекта **LinearLayout.LayoutParams**:

```
package com.example.viewapp;

```

```
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Gravity;
import android.widget.LinearLayout;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        LinearLayout linearLayout = new LinearLayout(this);
        linearLayout.setOrientation(LinearLayout.VERTICAL);
        LinearLayout.LayoutParams layoutParams = new LinearLayout.LayoutPara
                (LinearLayout.LayoutParams.WRAP_CONTENT, LinearLayout.Layou
// установка layout_gravity
        layoutParams.gravity = Gravity.CENTER;
        // первое текстовое поле
        TextView textView1 = new TextView(this);
        textView1.setText("Hello");
        textView1.setTextSize(30);
        linearLayout.addView(textView1, layoutParams);
        setContentView(linearLayout);
    }
}
```

В качестве значения передается одна из констант класса Gravity, которые аналогичны значениям атрибута.