

RelativeLayout

Последнее обновление:



RelativeLayout представляет объект `ViewGroup`, который располагает дочерние элементы относительно позиции других дочерних элементов разметки или относительно области самой разметки `RelativeLayout`. Используя относительное позиционирование, мы можем установить элемент по правому краю или в центре или иным способом, который предоставляет данный контейнер. Для установки элемента в файле `xml` мы можем применять следующие атрибуты:

- **`android:layout_above`**: располагает элемент над элементом с указанным `Id`
- **`android:layout_below`**: располагает элемент под элементом с указанным `Id`
- **`android:layout_toLeftOf`**: располагается слева от элемента с указанным `Id`
- **`android:layout_toRightOf`**: располагается справа от элемента с указанным `Id`
- **`android:layout_toStartOf`**: располагает начало текущего элемента, где начинается элемент с указанным `Id`
- **`android:layout_toEndOf`**: располагает начало текущего элемента, где завершается элемент с указанным `Id`

- **android:layout_alignBottom:** выравнивает элемент по нижней границе другого элемента с указанным Id
- **android:layout_alignLeft:** выравнивает элемент по левой границе другого элемента с указанным Id
- **android:layout_alignRight:** выравнивает элемент по правой границе другого элемента с указанным Id
- **android:layout_alignStart:** выравнивает элемент по линии, у которой начинается другой элемент с указанным Id
- **android:layout_alignEnd:** выравнивает элемент по линии, у которой завершается другой элемент с указанным Id
- **android:layout_alignTop:** выравнивает элемент по верхней границе другого элемента с указанным Id
- **android:layout_alignBaseline:** выравнивает базовую линию элемента по базовой линии другого элемента с указанным Id
- **android:layout_alignParentBottom:** если атрибут имеет значение true, то элемент прижимается к нижней границе контейнера
- **android:layout_alignParentRight:** если атрибут имеет значение true, то элемент прижимается к правому краю контейнера
- **android:layout_alignParentLeft:** если атрибут имеет значение true, то элемент прижимается к левому краю контейнера
- **android:layout_alignParentStart:** если атрибут имеет значение true, то элемент прижимается к начальному краю контейнера (при левосторонней ориентации текста - левый край)
- **android:layout_alignParentEnd:** если атрибут имеет значение true, то элемент прижимается к конечному краю контейнера (при левосторонней ориентации текста - правый край)
- **android:layout_alignParentTop:** если атрибут имеет значение true, то элемент прижимается к верхней границе контейнера
- **android:layout_centerInParent:** если атрибут имеет значение true, то элемент располагается по центру родительского контейнера
- **android:layout_centerHorizontal:** при значении true выравнивает элемент по центру по горизонтали

- **android:layout_centerVertical:** при значении true выравнивает элемент по центру по вертикали

Например, позиционирование относительно контейнера RelativeLayout:

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent">
6
7      <TextView android:text="Left Top"
8          android:layout_height="wrap_content"
9          android:layout_width="wrap_content"
10         android:textSize="26sp"
11         android:layout_alignParentLeft="true"
12         android:layout_alignParentTop="true" />
13
14     <TextView android:text="Right Top"
15         android:layout_height="wrap_content"
16         android:layout_width="wrap_content"
17         android:textSize="26sp"
18         android:layout_alignParentRight="true"
19         android:layout_alignParentTop="true" />
20
21     <TextView android:text="Left Bottom"
22         android:layout_height="wrap_content"
23         android:layout_width="wrap_content"
24         android:textSize="26sp"
25         android:layout_alignParentLeft="true"
26         android:layout_alignParentBottom="true" />
27
28     <TextView android:text="Right Bottom"
29         android:layout_height="wrap_content"
30         android:layout_width="wrap_content"
31         android:textSize="26sp"
32         android:layout_alignParentRight="true"
33         android:layout_alignParentBottom="true" />
34 </RelativeLayout>
```



Для позиционирования относительно другого элемента, нам надо указать id этого элемента. Так, поместим на RelativeLayout текстовое поле и кнопку:

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent">
6
7      <EditText
8          android:id="@+id/edit_message"
9          android:layout_width="match_parent"
10         android:layout_height="wrap_content"
11         android:layout_centerInParent="true"/>
12     <Button
13         android:layout_width="wrap_content"
14         android:layout_height="wrap_content"
15         android:text="Отправить"
16         android:layout_alignRight="@id/edit_message"
17         android:layout_below="@id/edit_message"
18     />
19 </RelativeLayout>
```

В данном случае поле EditText располагается по центру в RelativeLayout, а кнопка помещается под EditText и выравнивается по его правой границе:



Программное создание RelativeLayout

Создадим элемент RelativeLayout программно в коде Java:

```
1 package com.example.viewapp;
2
3 import androidx.appcompat.app.AppCompatActivity;
4 import android.os.Bundle;
5 import android.widget.Button;
6 import android.widget.EditText;
7 import android.widget.RelativeLayout;
8
9 public class MainActivity extends AppCompatActivity {
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14
15         RelativeLayout relativeLayout = new RelativeLayout(this);
16
17         EditText editText = new EditText(this);
18         editText.setId(editText.generateViewId());
19
20         Button button = new Button(this);
21         button.setText("Отправить");
22
23         // устанавливаем параметры положения для EditText
24         RelativeLayout.LayoutParams editTextParams = new RelativeLayout.Lay
25             RelativeLayout.LayoutParams.MATCH_PARENT,
```

```

26         RelativeLayout.LayoutParams.WRAP_CONTENT
27     );
28     // выравнивание по центру родительского контейнера
29     editTextParams.addRule(RelativeLayout.CENTER_IN_PARENT);
30     // добавляем в RelativeLayout
31     relativeLayout.addView(editText, editTextParams);
32
33     // устанавливаем параметры положения для Button
34     RelativeLayout.LayoutParams buttonParams = new RelativeLayout.LayoutParams(
35         RelativeLayout.LayoutParams.WRAP_CONTENT,
36         RelativeLayout.LayoutParams.WRAP_CONTENT
37     );
38     // выравнивание справа и снизу от поля EditText
39     buttonParams.addRule(RelativeLayout.BELOW, editText.getId());
40     buttonParams.addRule(RelativeLayout.ALIGN_RIGHT, editText.getId());
41     // добавляем в RelativeLayout
42     relativeLayout.addView(button, buttonParams);
43
44     setContentView(relativeLayout);
45 }
46 }

```

Чтобы задать положение элемента в контейнере, применяется класс

RelativeLayout.LayoutParams. Через конструктор устанавливаются значения для для ширины и высоты. Например, у элемента `EditText` для ширины устанавливается значение `MATCH_PARENT`, а для высоты - `WRAP_CONTENT`.

С помощью метода **addRule()** мы можем добавлять дополнительные правила для позиционирования элемента. Этот метод в качестве параметра принимает числовую константу, которая представляет параметр позиционирования и которая аналогична атрибуту. Например, атрибуту `android:layout_centerInParent` будет соответствовать константа `CENTER_IN_PARENT`, а атрибуту `android:layout_alignRight` константа `ALIGN_RIGHT`.

Стоит отметить, что в целях упрощения кода для установки `id` у `EditText` вызывается метод **generateViewId()**, который позволяет программно сгенерировать `id` для элемента управления.

Затем установленный `id` передается в качестве второго параметра в метод `addRule` при установке правил для кнопки:

```

1 buttonParams.addRule(RelativeLayout.BELOW, editText.getId());

```

Тем самым мы указываем относительно какого элемента надо задать расположение.