GridLayout

Последнее обновление:

GridLayout представляет еще один контейнер, который позволяет создавать табличные представления. GridLayout состоит из коллекции строк, каждая из которых состоит из отдельных ячеек:

```
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:rowCount="3"
    android:columnCount="3">

    <Button android:text="1" />
    <Button android:text="2" />
    <Button android:text="3" />
    <Button android:text="4" />
    <Button android:text="5" />
    <Button android:text="6" />
    <Button android:text="7" />

    <Button android:text="8" />

    <Button android:text="9" />
</GridLayout>
```

С помощью атрибутов **android:rowCount** и **android:columnCount** устанавливается число строк и столбцов соответственно. Так, в данном случае устанавливаем 3 строки и 3 столбца. GridLayout автоматически может позиционировать вложенные элементы управления по строкам. Так, в нашем случае первая кнопка попадает в первую ячейку (первая строка первый столбец), вторая кнопка - во вторую ячейку и так далее.

При этом ширина столбцов устанавливается автоматически по ширине самого широкого элемента.

Однако мы можем явно задать номер столбца и строки для определенного элемента, а при необходимости растянуть на несколько столбцов или строк. Для этого мы можем применять следующие атрибуты:

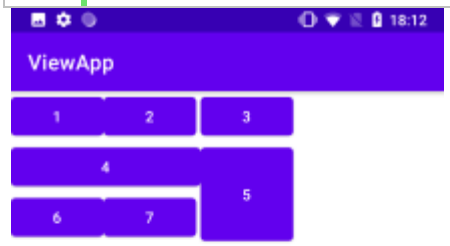
- **android:layout_column**: номер столбца (отсчет идет от нуля)
- **android:layout_row**: номер строки
- **android:layout_columnSpan**: количество столбцов, на которые растягивается элемент
- **android:layout_rowSpan**: количество строк, на которые растягивается элемент

Например:

```
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:rowCount="3"
```

```
    android:columnCount="3">

    <Button
        android:text="1"
        android:layout_column="0"
        android:layout_row="0" />
    <Button android:text="2"
        android:layout_column="1"
        android:layout_row="0"/>
    <Button android:text="3"
        android:layout_column="2"
        android:layout_row="0"  />
    <Button android:text="4"
        android:layout_width="180dp"
        android:layout_columnSpan="2"/>
    <Button android:text="5"
        android:layout_height="100dp"
        android:layout_rowSpan="2"/>
    <Button android:text="6" />
    <Button android:text="7"/>
</GridLayout>
```



Программное создание GridLayout

Среди методов **GridLayout** следует отметить методы **setRowCount()** и **setColumnCount()**, которые позволяют задать соответственно количество строк и столбцов. Например, определим в коде GridLayout, аналогичнй первому примеру в статье:

```
package com.example.viewapp;

```

```
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Gravity;
import android.widget.Button;
import android.widget.EditText;
import android.widget.GridLayout;
import android.widget.LinearLayout;
import android.widget.TableLayout;
import android.widget.TableRow;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

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

        GridLayout gridLayout = new GridLayout( this);
        // количество строк
        gridLayout.setRowCount(3);
        // количество столбцов
        gridLayout.setColumnCount(3);

        for(int i = 1; i <=9; i++){
            Button btn = new Button(this);
            btn.setText(String.valueOf(i));
            gridLayout.addView(btn);
        }
        setContentView(gridLayout);
    }
}
```

В данном случае GridLayout имеет три строки и три столбца. При добавлении виджетов (в данном случае кнопок) они последовательно помещаются в ячейки грида по одному виджету в ячейке.

GridLayout.LayoutParams

Для более детальной настройки расположения виджета в гриде можно использовать класс **GridLayout.LayoutParams**. Этот класс имеет ряд свойств, которые позволяют настроить расположение:

- **columnSpec**: задает столбец для расположения в виде объекта **GridLayout.Spec**
- **rowSpec**: задает строку для расположения в виде объекта **GridLayout.Spec**
- **leftMargin**: задает отступ слева
- **rightMargin**: задает отступ справа

- **topMargin**: задает отступ сверху
- **bottomMargin**: задает отступ снизу
- **width**: задает ширину виджета
- **height**: задает высоту виджета

Объект **GridLayout.Spec** позволяет задать размещение в ячейках столбца или строки. Для создание этого объекта применяется статический метод **GridLayout.spec()**, который имеет ряд версий. Отметим среди них следующие:

- `GridLayout.spec(int)`: задает столбец или строку, где располагается виджет. Отсчет ячеек начинается с нуля. Виджет занимает только одну ячейку
- `GridLayout.spec(int, int)`: первый параметр задает столбец или строку, где располагается виджет. Второй параметр указывает, насколько ячеек растягивается виджет
- `GridLayout.spec(int, android.widget.GridLayout.Alignment)`: первый параметр задает столбец или строку, где располагается виджет. Второй параметр устанавливает выравнивание виджета
- `GridLayout.spec(int, int, android.widget.GridLayout.Alignment)`: первый параметр задает столбец или строку, где располагается виджет. Второй параметр указывает, насколько ячеек растягивается виджет. Третий параметр устанавливает выравнивание виджета

Пример применения GridLayout.LayoutParams:

```
Button btn = new Button(this);
btn.setText("нажми");
GridLayout.LayoutParams layoutParams = new GridLayout.LayoutParams();
 // кнопка помещается в нулевой столбец и растягивается на 2 столбца
layoutParams.columnSpec = GridLayout.spec(0,2);
 // кнопка помещается во вторую строку и растягивается на 1 строку
layoutParams.rowSpec = GridLayout.spec(1,1);
layoutParams.leftMargin=5;
layoutParams.rightMargin=5;
layoutParams.topMargin=4;
layoutParams.bottomMargin=4;
layoutParams.width = GridLayout.LayoutParams.MATCH_PARENT;
layoutParams.height = GridLayout.LayoutParams.WRAP_CONTENT;
gridLayout.addView(btn, layoutParams);
```

Например, реализуем в коде второй пример из данной статьи:

```java
package com.example.viewapp;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.util.TypedValue;
import android.view.Gravity;
import android.widget.Button;
import android.widget.EditText;
import android.widget.GridLayout;
import android.widget.LinearLayout;
import android.widget.TableLayout;
import android.widget.TableRow;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

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

        GridLayout gridLayout = new GridLayout( this);

        // количество строк
        gridLayout.setRowCount(3);
        // количество столбцов
        gridLayout.setColumnCount(3);

        for(int i = 1; i <=3; i++){
            Button btn = new Button(this);
            btn.setText(String.valueOf(i));
            gridLayout.addView(btn);
        }

        Button btn4 = new Button(this);
        btn4.setText("4");
        GridLayout.LayoutParams layoutParams4 = new GridLayout.LayoutParams
        layoutParams4.columnSpec = GridLayout.spec(0,2);
        layoutParams4.width = (int) TypedValue.applyDimension(
                TypedValue.COMPLEX_UNIT_DIP, 180, getResources().getDisplayM
        gridLayout.addView(btn4, layoutParams4);


        Button btn5 = new Button(this);
        btn5.setText("5");
        GridLayout.LayoutParams layoutParams5 = new GridLayout.LayoutParams
        layoutParams5.rowSpec = GridLayout.spec(1,2);
        layoutParams5.height = (int) TypedValue.applyDimension(
                TypedValue.COMPLEX_UNIT_DIP, 100, getResources().getDisplayM
        gridLayout.addView(btn5, layoutParams5);
```

```
50
51          Button btn6 = new Button(this);
52          btn6.setText("6");
53          Button btn7 = new Button(this);
54          btn7.setText("7");
55          gridLayout.addView(btn6);
56          gridLayout.addView(btn7);
57
58          setContentView(gridLayout);
59      }
60  }
```