



*Белорусский государственный университет
информатики и радиоэлектроники*

Эргономика мобильных приложений

Преподаватель: Меженная Марина Михайловна

К.Т.Н., доцент,

доцент кафедры инженерной психологии и эргономики
а 609-2

mezhennaya@bsuir.by



Кафедра инженерной психологии и эргономики

Лекция 3: Графическое представление Activity: элементы экрана View и их свойства.

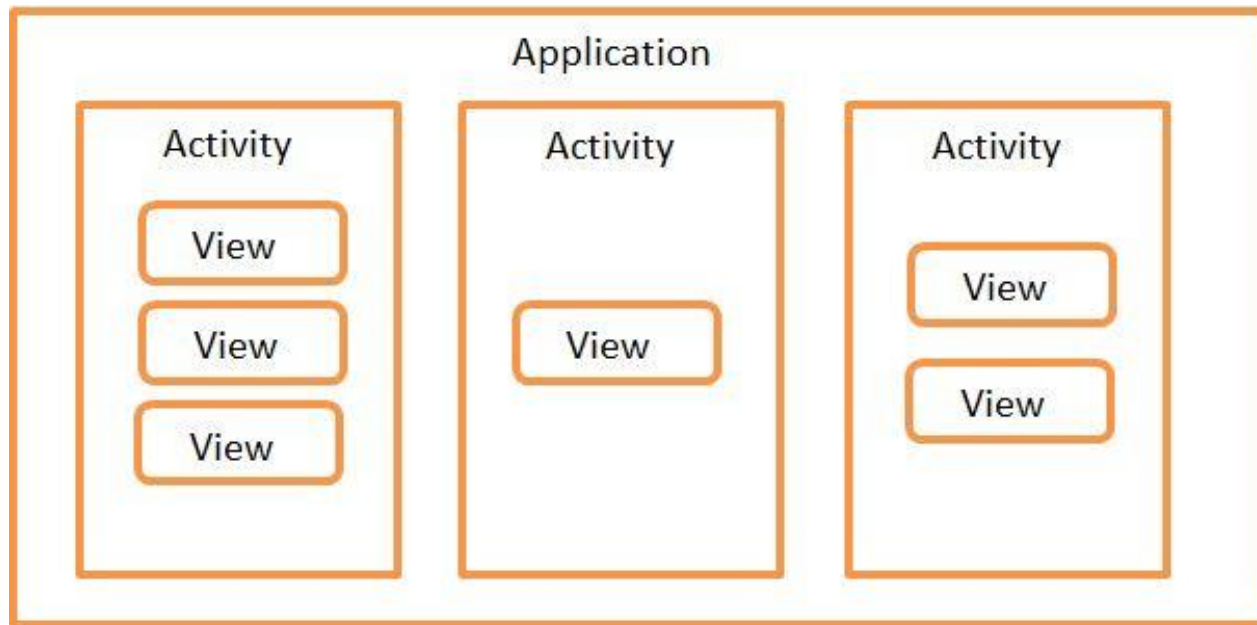
План лекции:

- 1.Графическое представление Activity: View и ViewGroup. Layout-файл.
- 2.Виды Layout.
- 3.Параметры элементов экрана View.
- 4.Смена ориентации экрана.

Графическое представление Activity

Приложение состоит из окон, называемых Activity.

Содержимое Activity формируется из различных компонентов, называемых View.



Графическое представление Activity: View и ViewGroup

В Android приложении графический интерфейс пользователя формируется с использованием объектов View (представление) и ViewGroup (группа представлений).

Самый распространенный пример ViewGroup – это Layout. Layout бывает различных типов и отвечает за то, как будут расположены его дочерние View на экране (таблицей, строкой, столбцом).

Виджеты - это простейшие (базовые) объекты View, которые служат интерфейсом для взаимодействия с пользователем. Примеры виджетов: кнопки, текстовые поля, чек-боксы.

Графическое представление Activity: res > layout > activity_main.xml

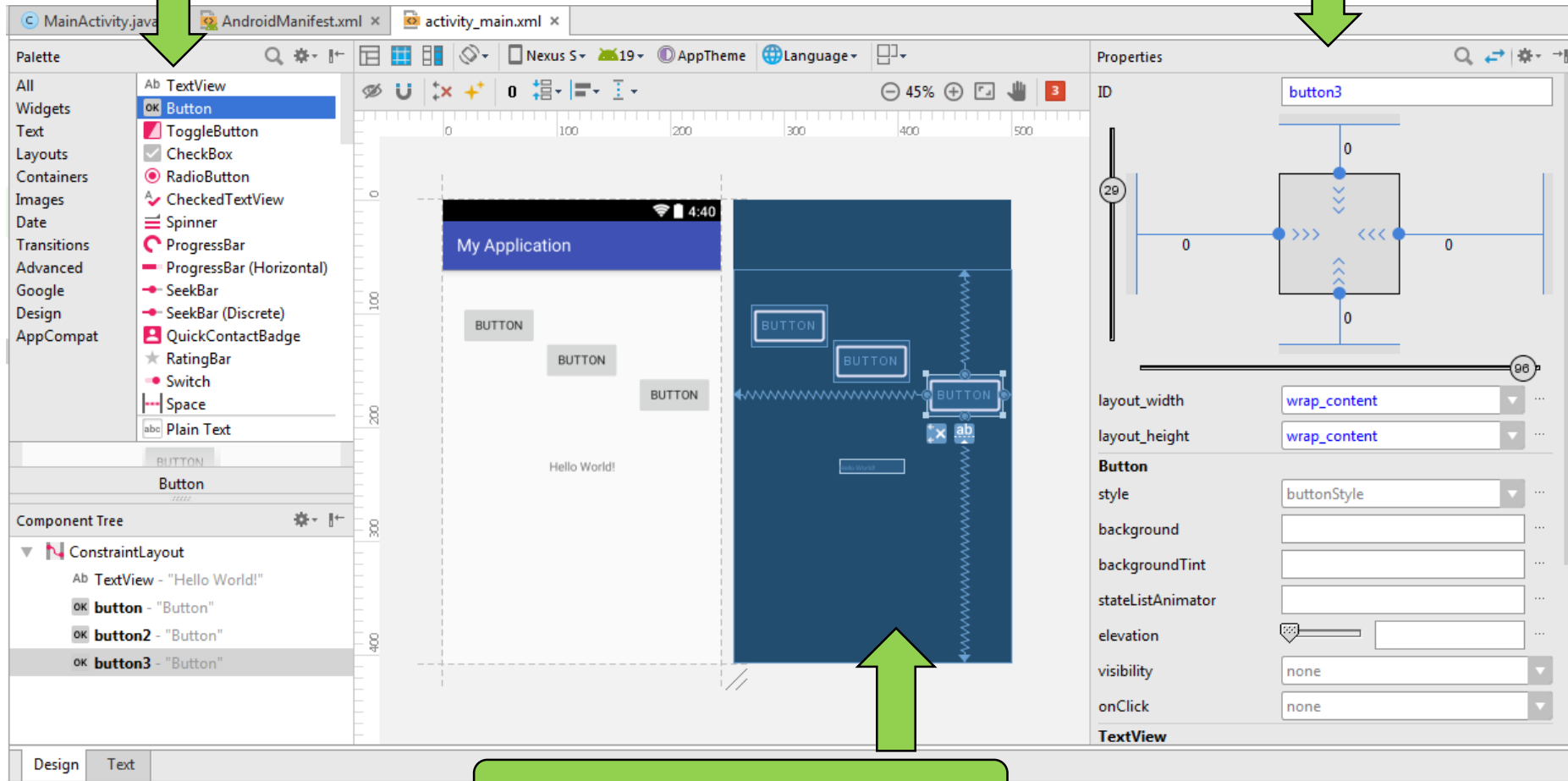
Разметка - это архитектура расположения элементов интерфейса пользователя для конкретного окна, представляющего деятельность.

Определяют разметку в layout-файлах с расширением xml.

Работать с графической разметкой можно с помощью конструктора (вкладка Design) или непосредственно в коде xml-файла (вкладка Text).

Графическое представление Activity: activity_main.xml, вкладка Design

1. Выбираем элементы интерфейса

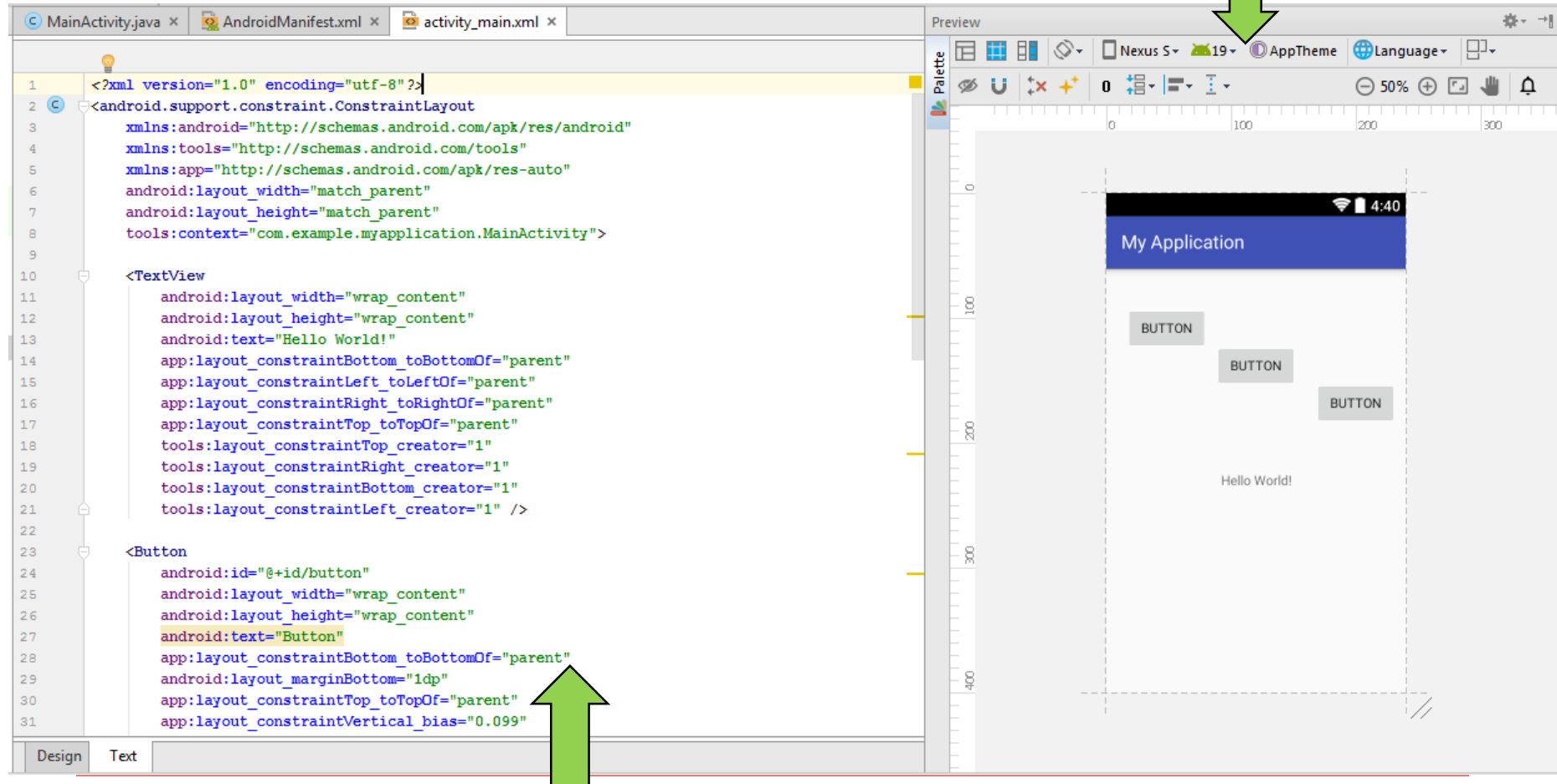


3. Настраиваем свойства

2. Размещаем их на layout

Графическое представление Activity: activity_main.xml, вкладка Text

Смотрим результат



При необходимости редактируем код

Виды Layout

Расположение View-элементов на экране зависит от ViewGroup (Layout), в которой они находятся. Сами Layout-ы также можно группировать (вкладывать один в другой и т.д.).

Рассмотрим основные виды Layout:

➔ **LinearLayout** – отображает View-элементы в виде одной строки (если он Horizontal) или одного столбца (если он Vertical).

TableLayout – отображает элементы в виде таблицы, по строкам и столбцам.

RelativeLayout – для каждого элемента настраивается его положение относительно других элементов.

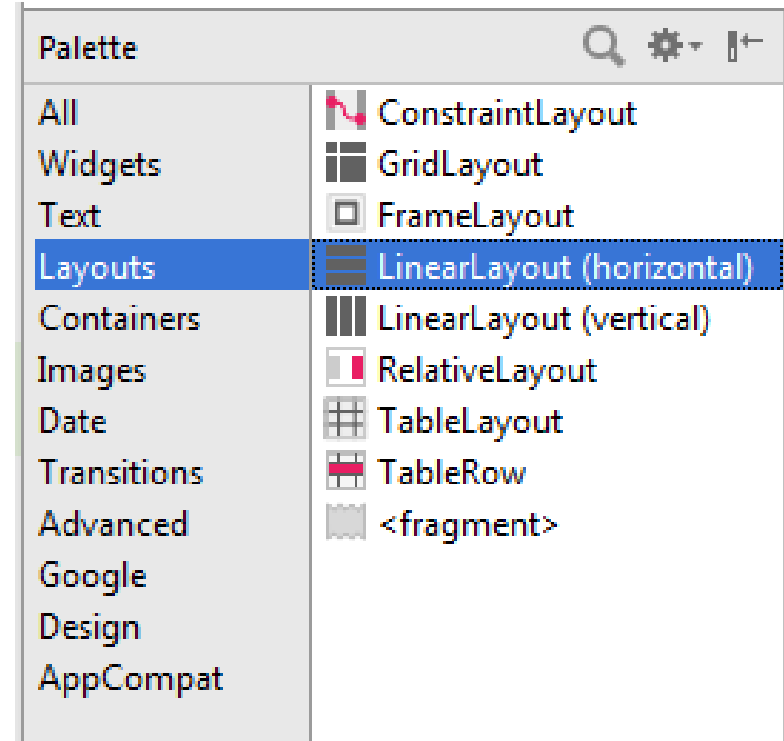
➔ **ConstraintLayout** – введен Android в 2016 году; в версии Android Studio 2.3 установлен по умолчанию как корневой.

Виды Layout: LinearLayout (LL)

Этот вид ViewGroup по умолчанию предлагается при создании новых layout-файлов.

LL имеет свойство Orientation, которое определяет, как будут расположены дочерние элементы — горизонтальной или вертикальной линией.

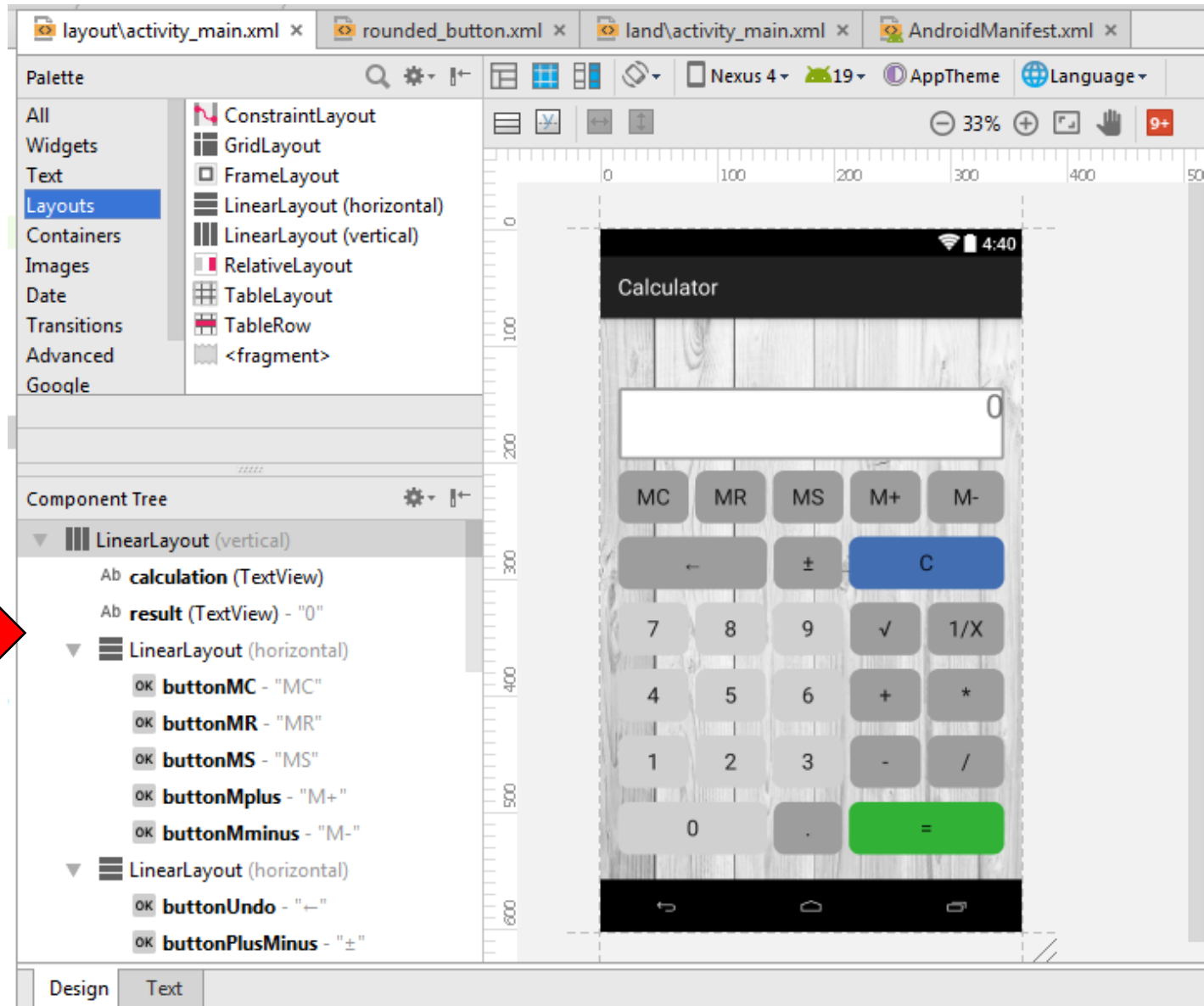
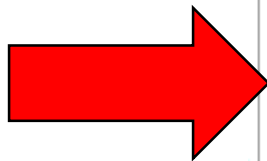
Путем вложения в один LL других LL можно создавать экраны различной сложности.



Виды Layout: LinearLayout (LL)

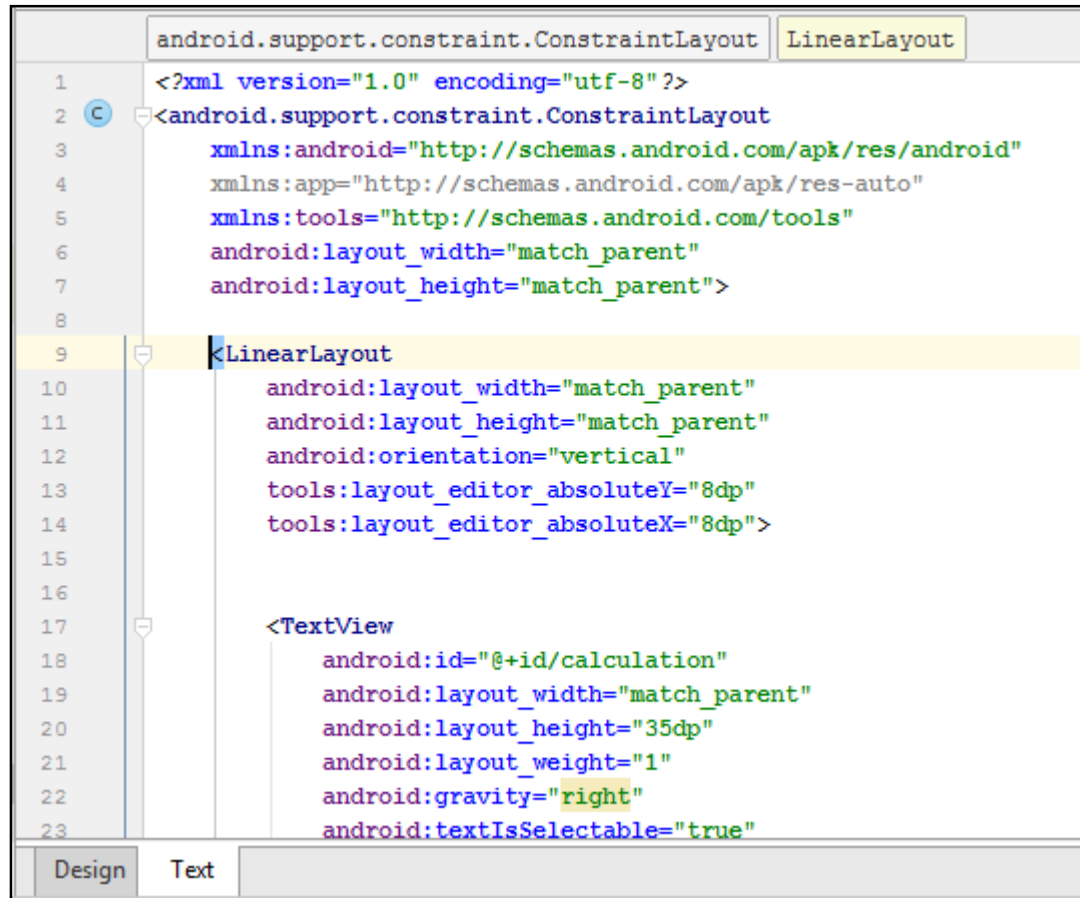
вкладка Design

Формируем
вложенную
структуру
LL



Виды Layout: LinearLayout (LL)

вкладка Text



The screenshot shows the XML editor in Android Studio. At the top, there are two tabs: 'android.support.constraint.ConstraintLayout' and 'LinearLayout'. The 'LinearLayout' tab is selected and highlighted in yellow. The XML code is as follows:

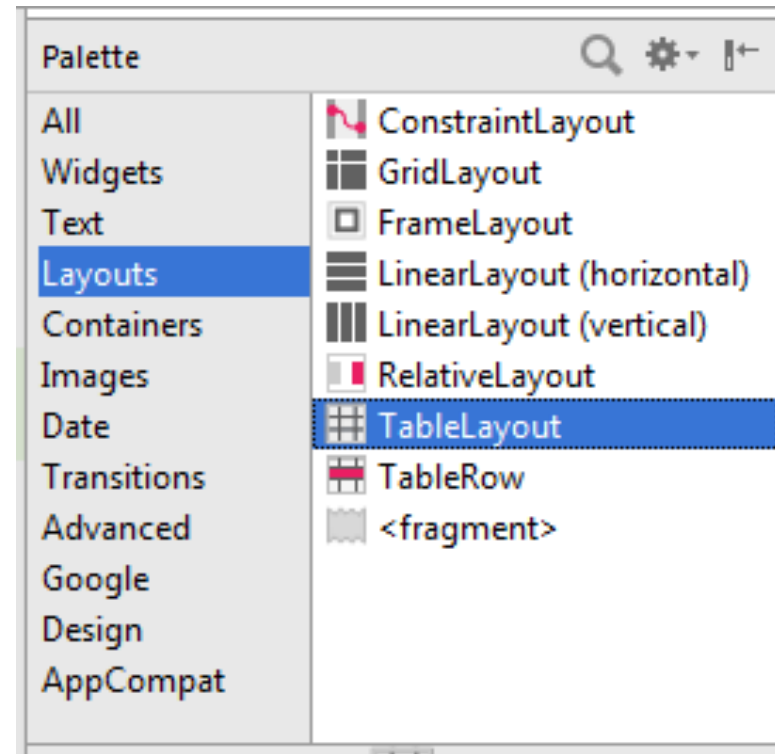
```
1 <?xml version="1.0" encoding="utf-8"?>
2 <android.support.constraint.ConstraintLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:app="http://schemas.android.com/apk/res-auto"
5     xmlns:tools="http://schemas.android.com/tools"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent">
8
9     <LinearLayout
10         android:layout_width="match_parent"
11         android:layout_height="match_parent"
12         android:orientation="vertical"
13         tools:layout_editor_absoluteY="8dp"
14         tools:layout_editor_absoluteX="8dp">
15
16
17         <TextView
18             android:id="@+id/calculation"
19             android:layout_width="match_parent"
20             android:layout_height="35dp"
21             android:layout_weight="1"
22             android:gravity="right"
23             android:textIsSelectable="true"
```

At the bottom of the editor, there are two tabs: 'Design' and 'Text'. The 'Text' tab is selected and highlighted in yellow.

Виды Layout: TableLayout (TL)

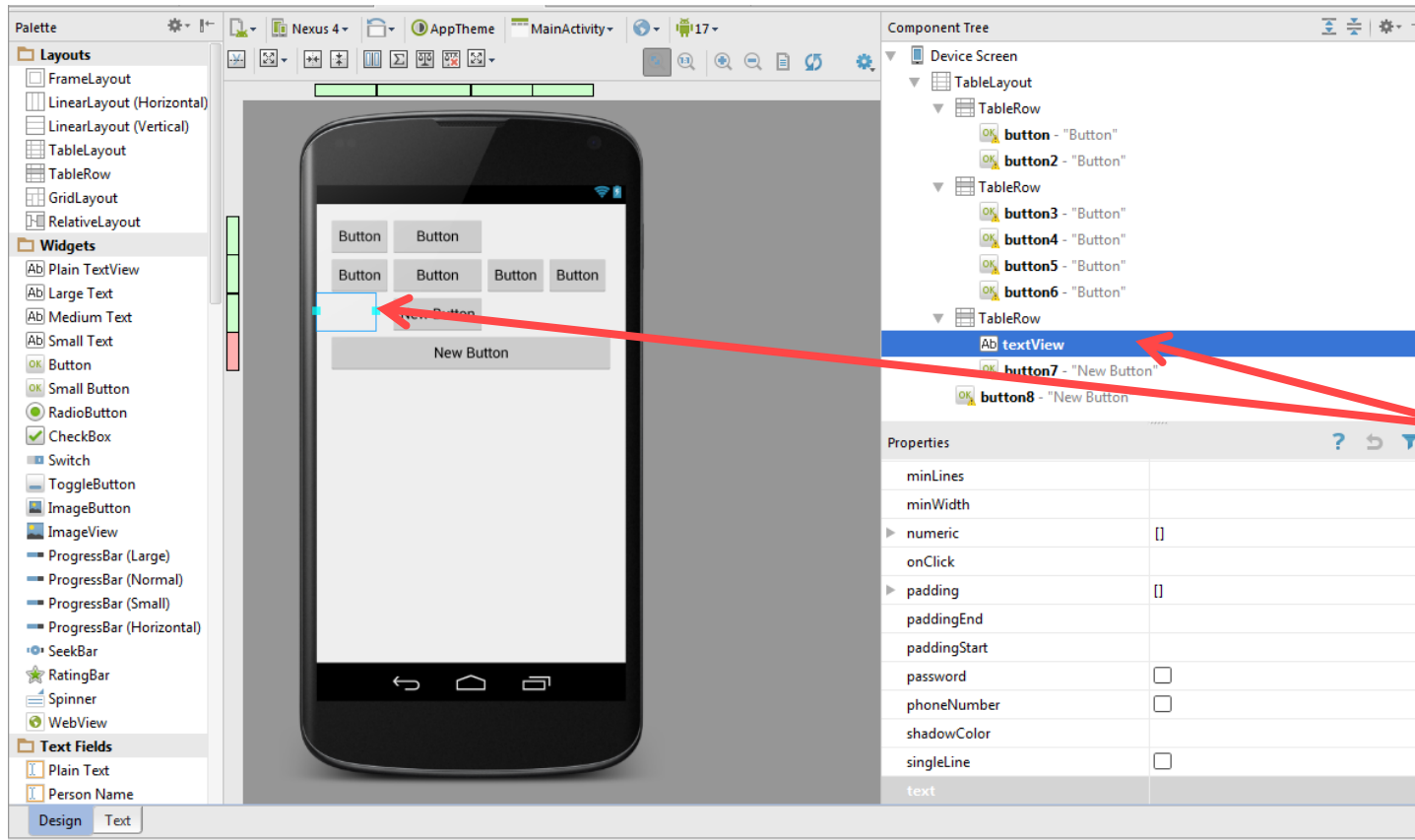
TL состоит из строк TableRow (TR) или из View (ViewGroup) в качестве строки.

Каждая TR содержит View-элементы, формирующие столбцы. Но кол-во столбцов в таблице должно быть равным для всех строк. Поэтому, если в разных TR разное кол-во View-элементов (столбцов), то общее кол-во столбцов определяется по TR с максимальным кол-вом.



Виды Layout: TableLayout (TL)

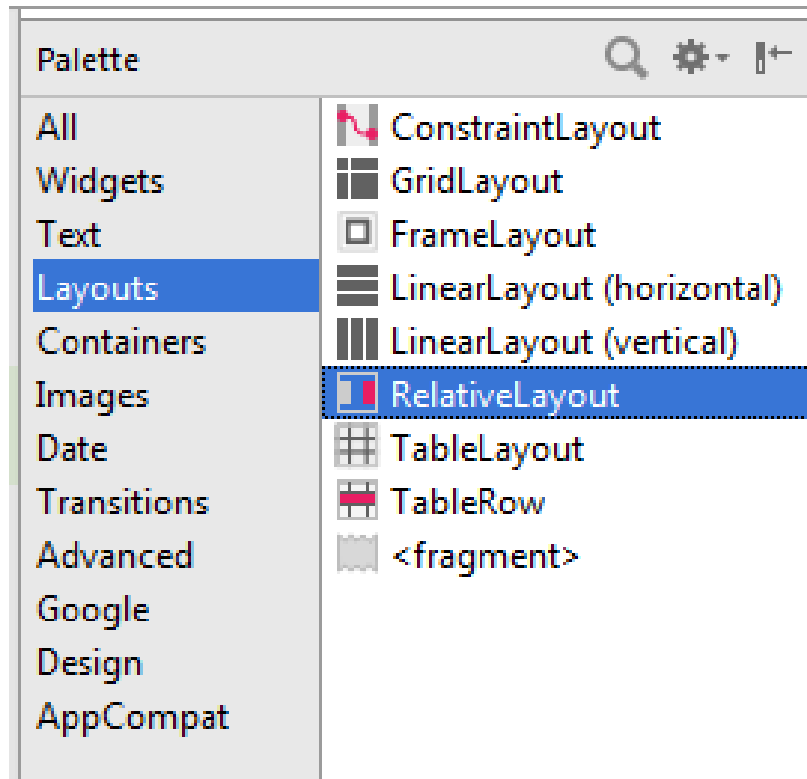
вкладка Design



Для создания
«пустого»
пространства
используйте
TextView

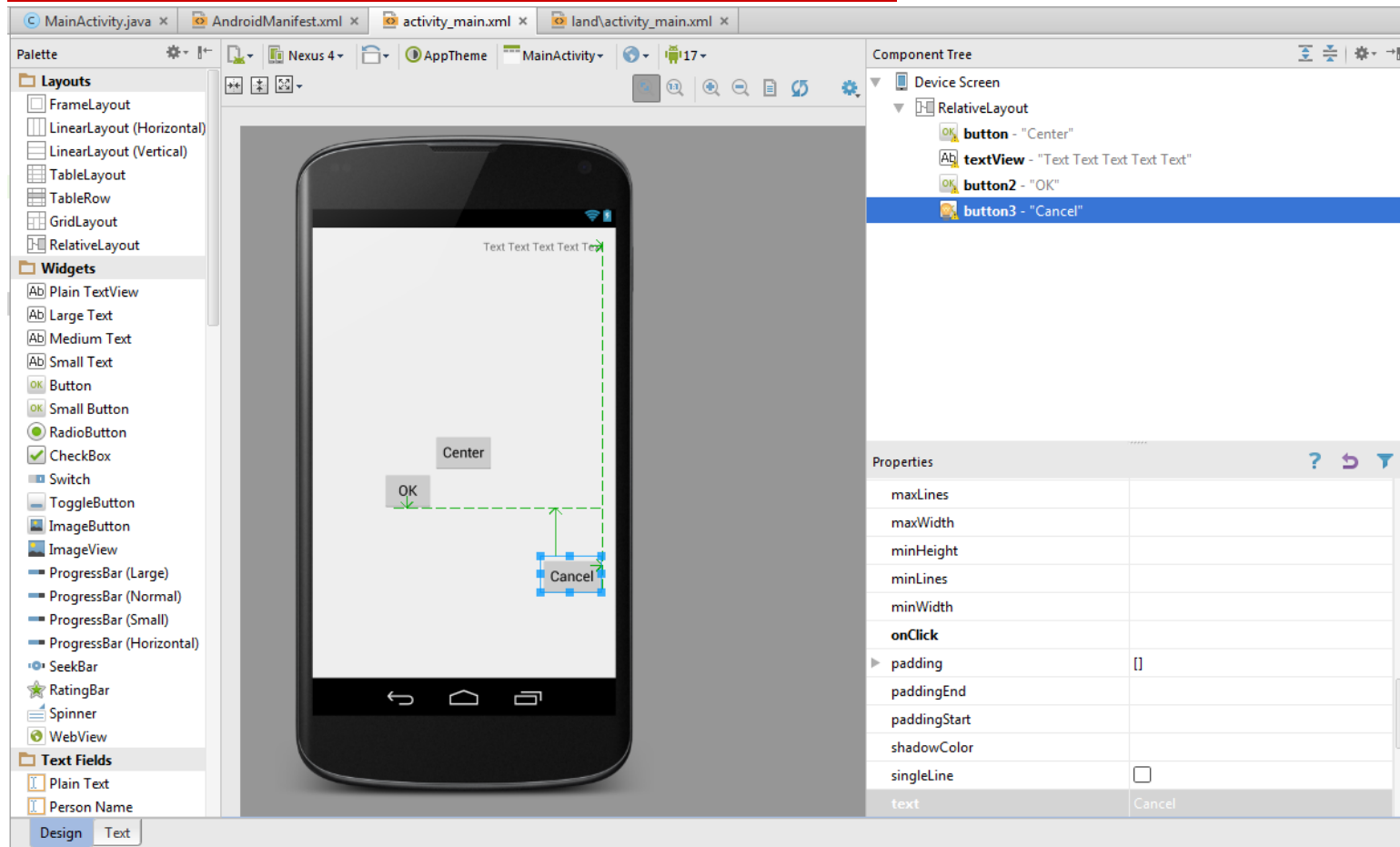
Виды Layout: RelativeLayout (RL)

В этом виде Layout каждый View-элемент может быть расположен определенным образом относительно указанного View-элемента.



Виды Layout: RelativeLayout (RL)

вкладка Design



Виды Layout: RelativeLayout (RL)

Виды отношений:

- 1) слева, справа, сверху, снизу указанного элемента (`layout_toLeftOf`, `layout_toRightOf`, `layout_above`, `layout_below`)
 - 2) выравненным по левому, правому, верхнему, нижнему краю указанного элемента (`layout_alignLeft`, `layout_alignRight`, `layout_alignTop`, `layout_alignBottom`)
 - 3) выравненным по левому, правому, верхнему, нижнему краю родителя (`layout_alignParentLeft`, `layout_alignParentRight`, `layout_alignParentTop`, `layout_alignParentBottom`)
 - 4) выравненным по центру вертикально, по центру горизонтально, по центру вертикально и горизонтально относительно родителя (`layout_centerVertical`, `layout_centerHorizontal`, `layout_centerInParent`)
-

Виды Layout: RelativeLayout (RL)

вкладка Text

MainActivity.java x AndroidManifest.xml x activity_main.xml x land\activity_main.xml x Preview

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".MainActivity"
    android:orientation="vertical">

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Center"
        android:id="@+id/button"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Text Text Text Text Text"
        android:id="@+id/textView"
        android:layout_alignParentTop="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="OK"
        android:id="@+id/button2"
        android:layout_below="@+id/button"
        android:layout_toLeftOf="@+id/button"
        android:layout_toStartOf="@+id/button" />

    <Button
        android:layout_width="wrap_content"
```

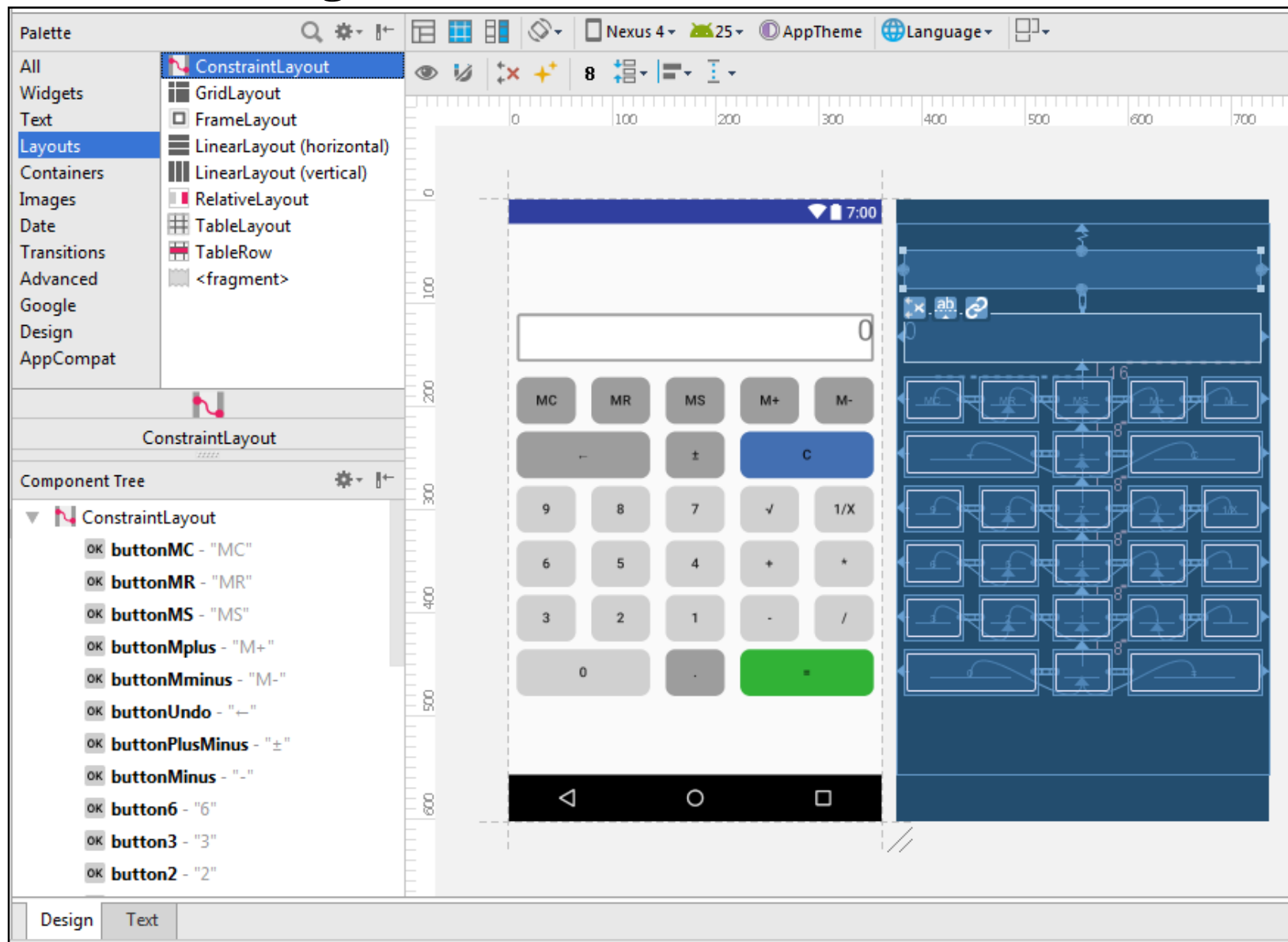
Design Text

Preview

Nexus 4+ AppTheme MainActivity 17

Виды Layout: ConstraintLayout

вкладка Design



Виды Layout: ConstraintLayout

вкладка Text

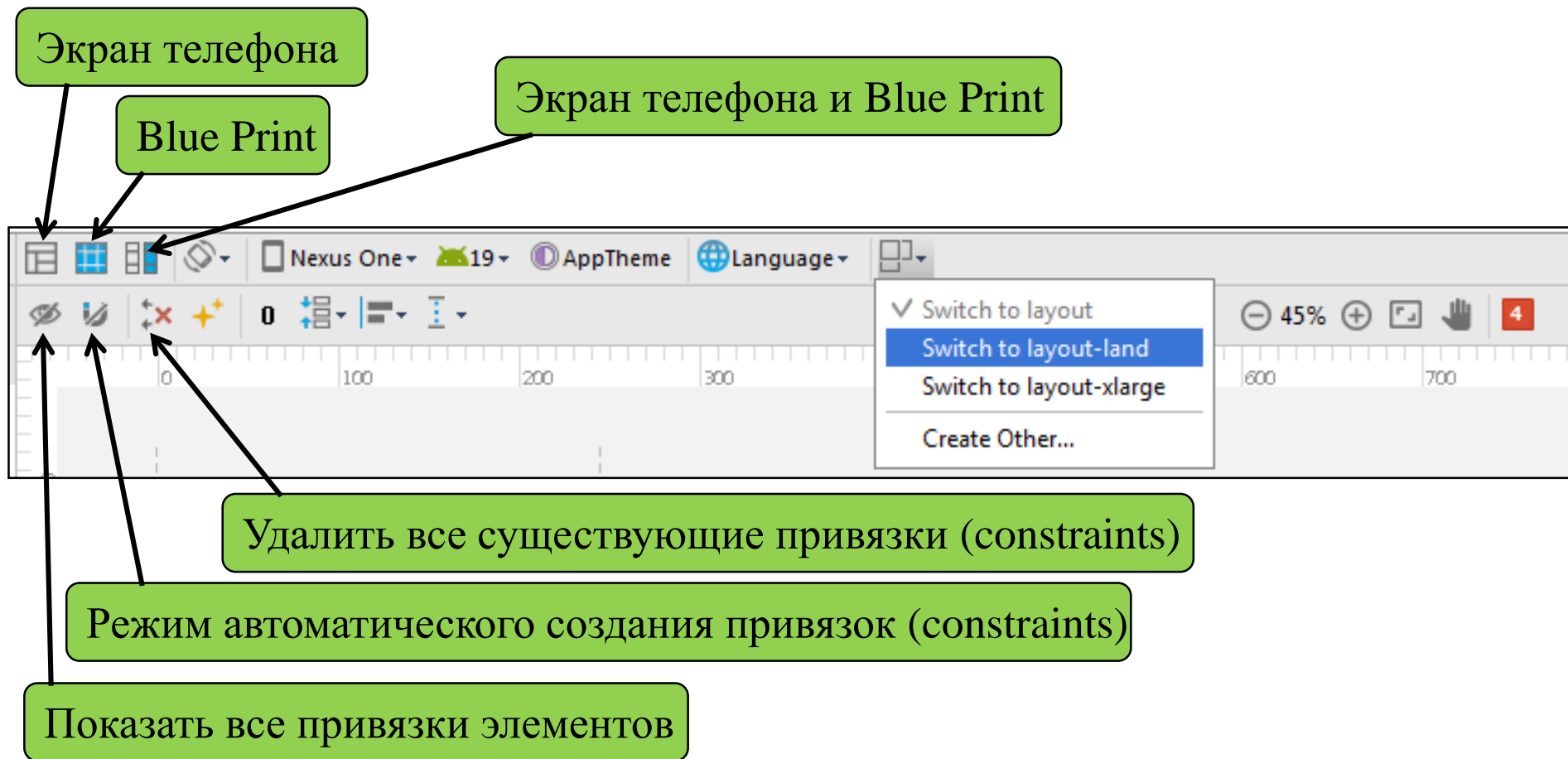
Корневой
Layout

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.myapplication.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:layout_constraintTop_creator="1"
        tools:layout_constraintRight_creator="1"
        tools:layout_constraintBottom_creator="1"
        tools:layout_constraintLeft_creator="1" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        app:layout_constraintBottom_toBottomOf="parent"
        android:layout_marginBottom="1dp"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.099" />
</android.support.constraint.ConstraintLayout>
```

Панель инструментов для работы с графическим представлением приложения



Панель инструментов для работы с графическим представлением приложения

Посмотреть при различной ориентации экрана

Посмотреть на различных по размеру экранах

Изменить текущий API

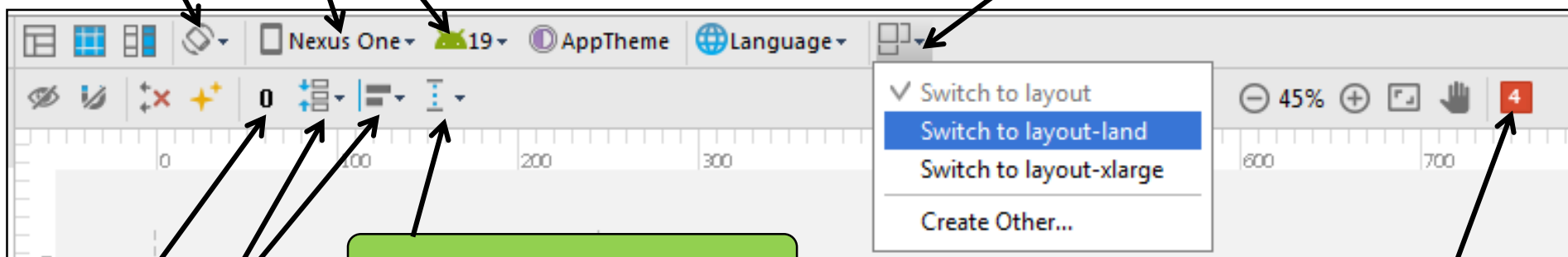
Создать xml-файл
гориз.ориент./под планшет

Добавить линии сетки

Связывание в цепи, выравнивание элементов

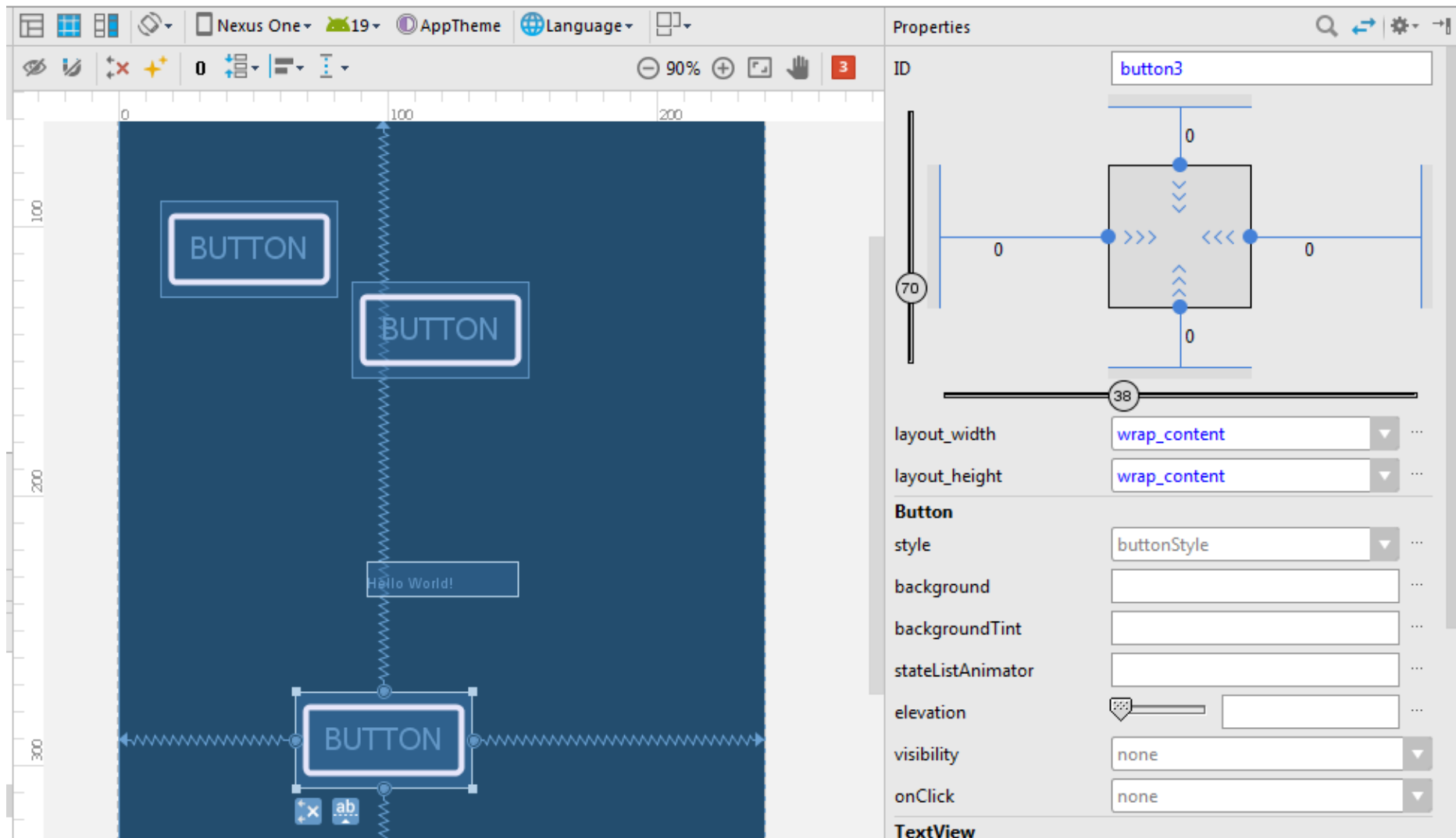
Настроить список отступов

Warnings/errors: список



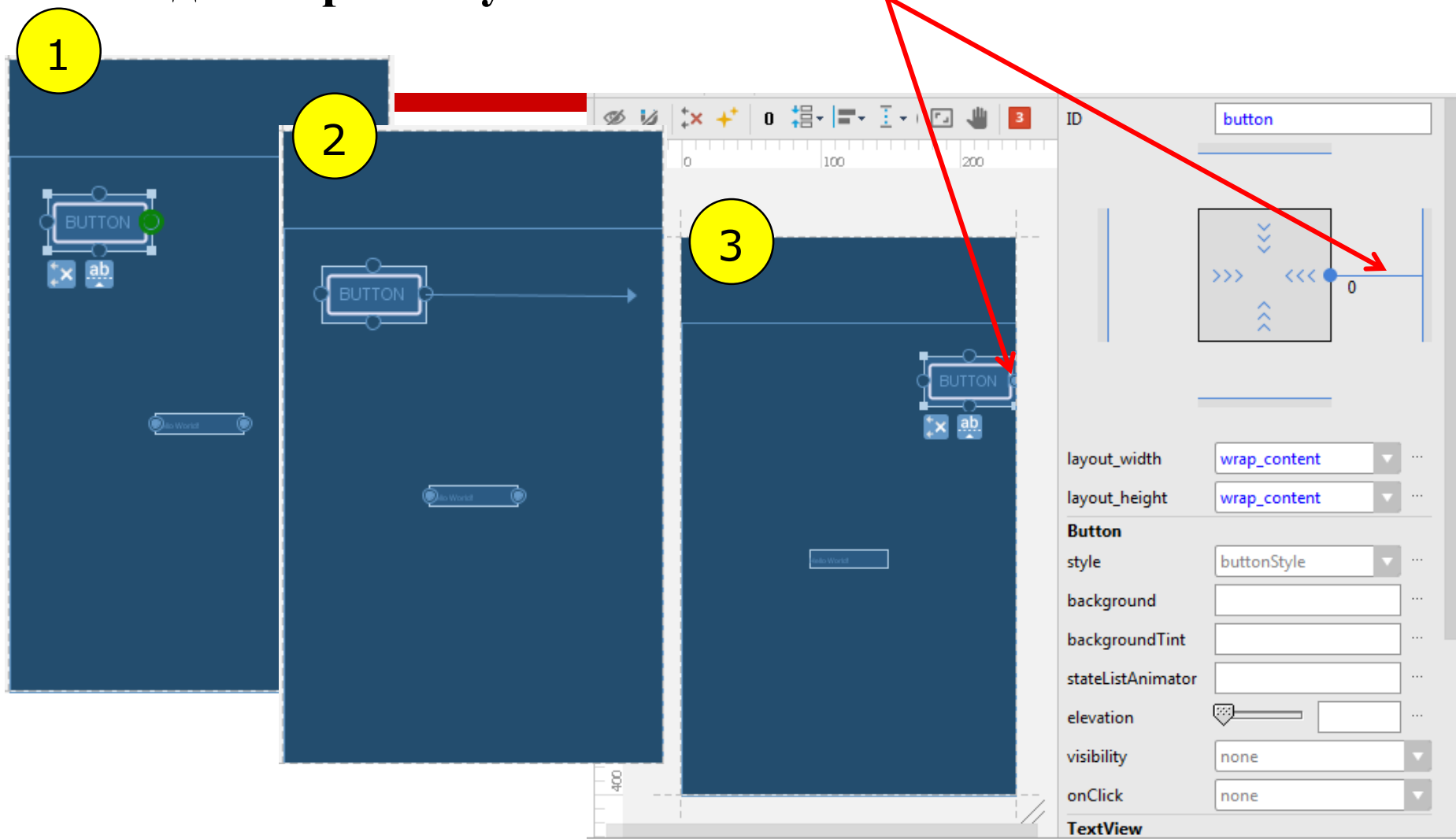
Виды Layout: ConstraintLayout

вкладка Design



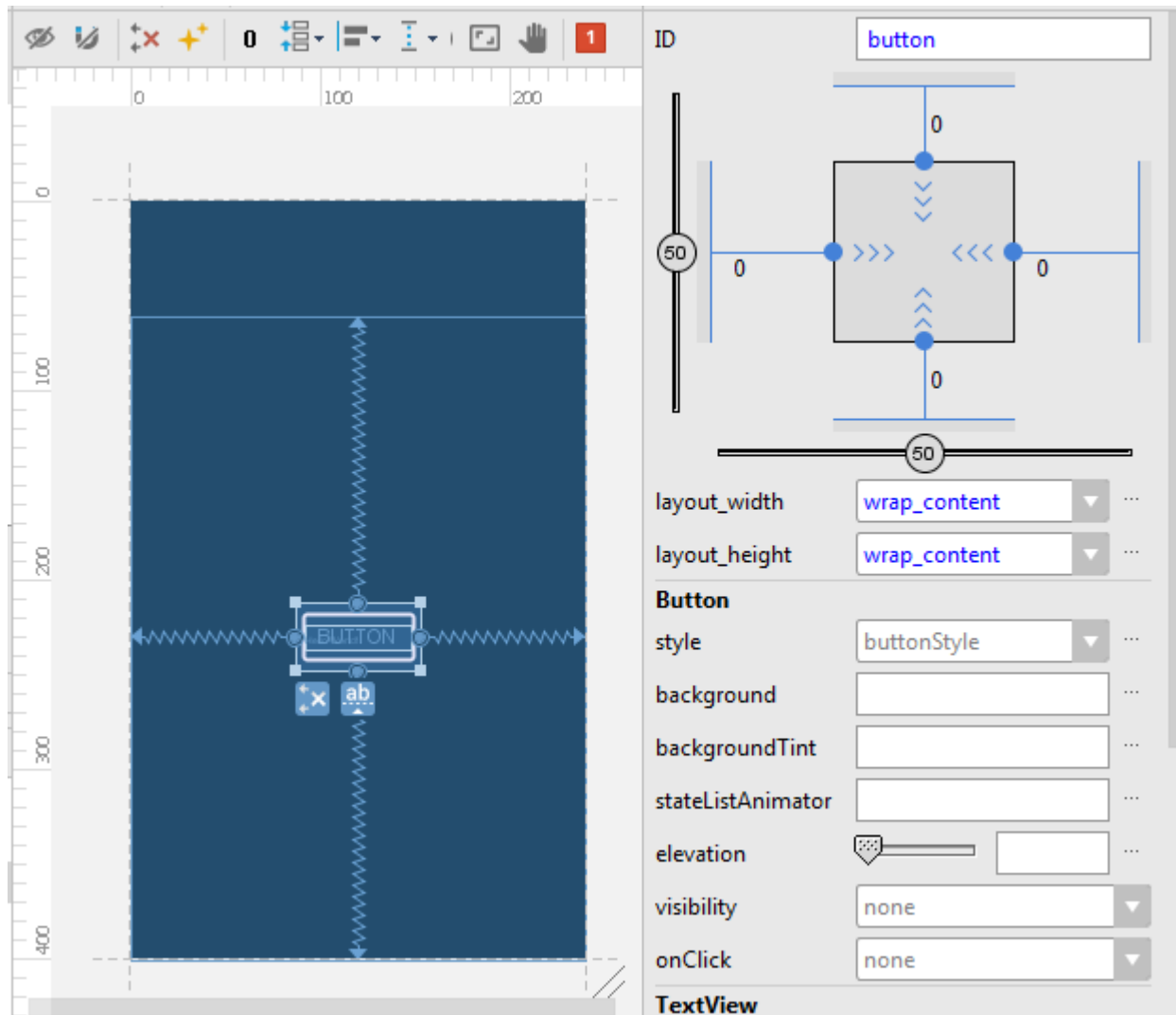
Виды Layout: ConstraintLayout

Создаем привязку



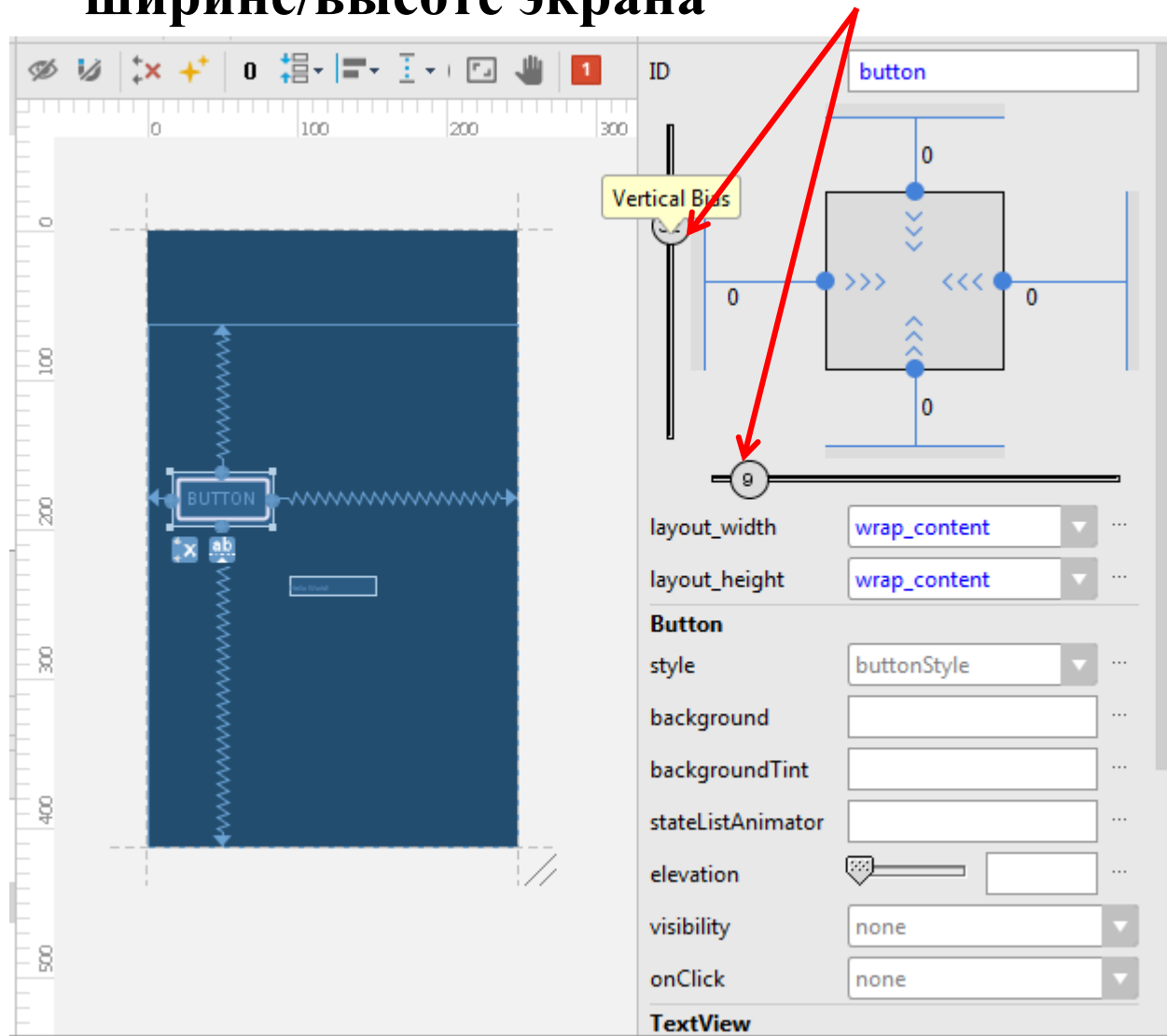
Виды Layout: ConstraintLayout

Привязки сделаны



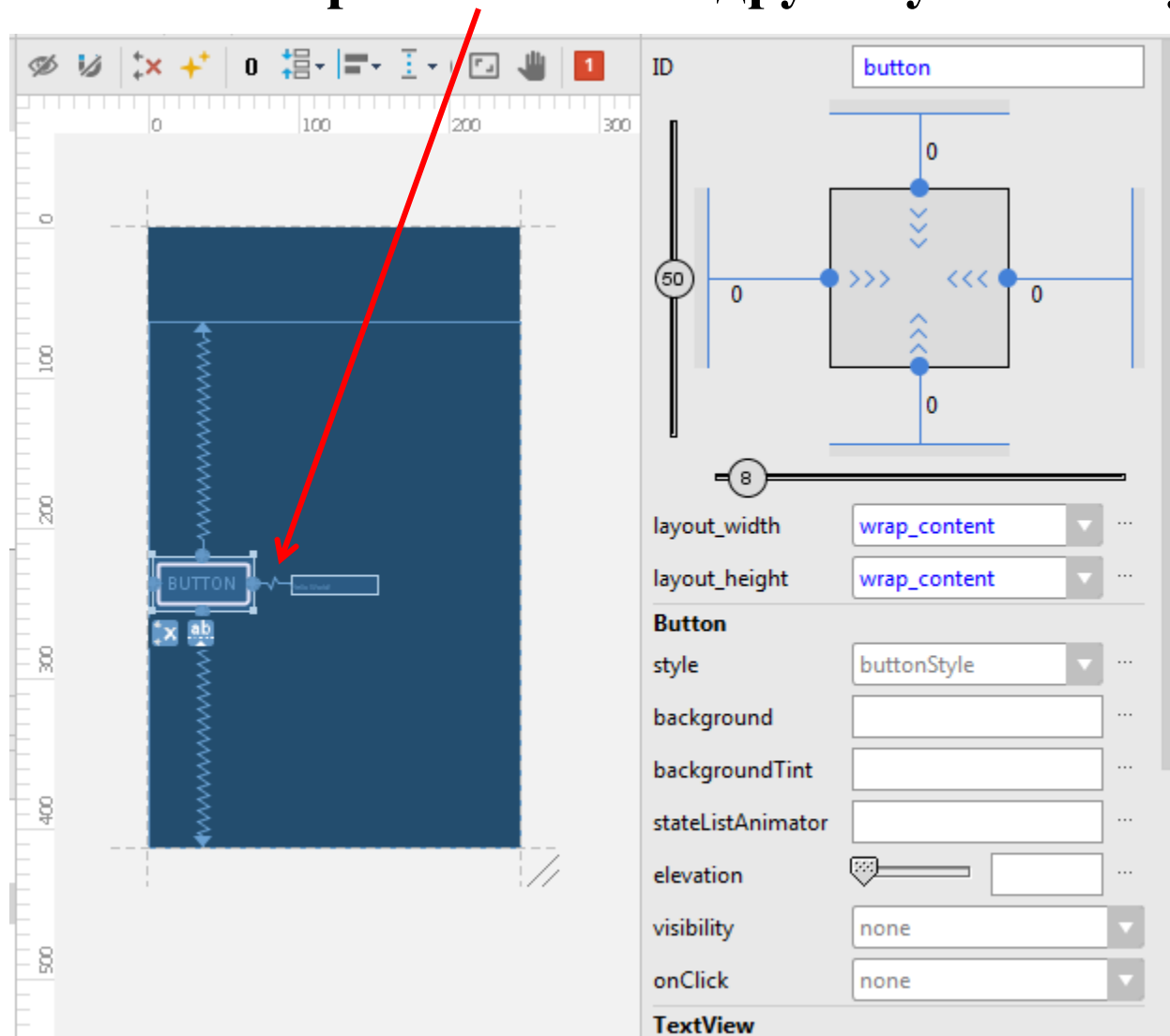
Виды Layout: ConstraintLayout

Можно двигать элемент в процентном отношении к ширине/высоте экрана

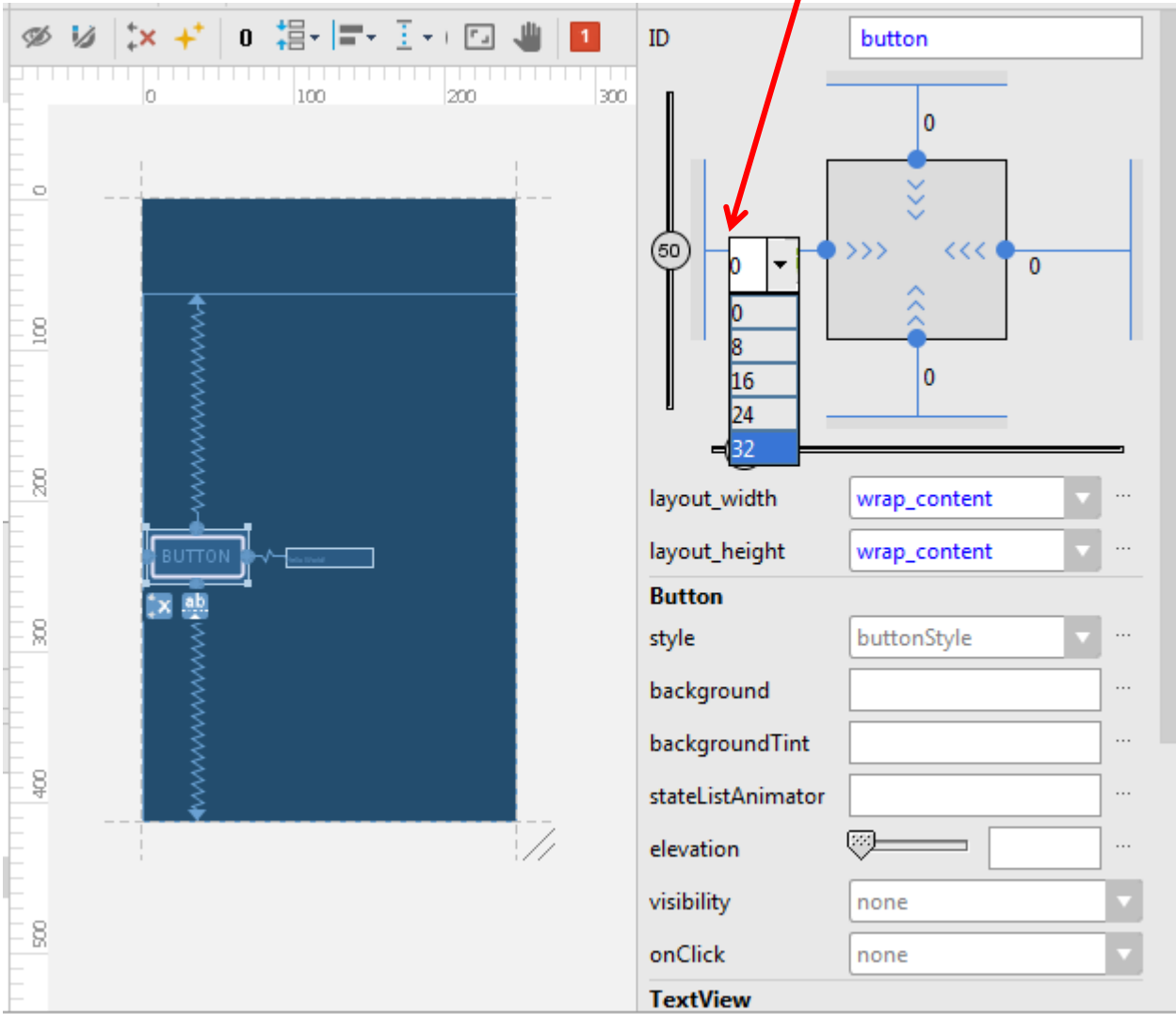


Виды Layout: ConstraintLayout

Можно привязывать к другому элементу

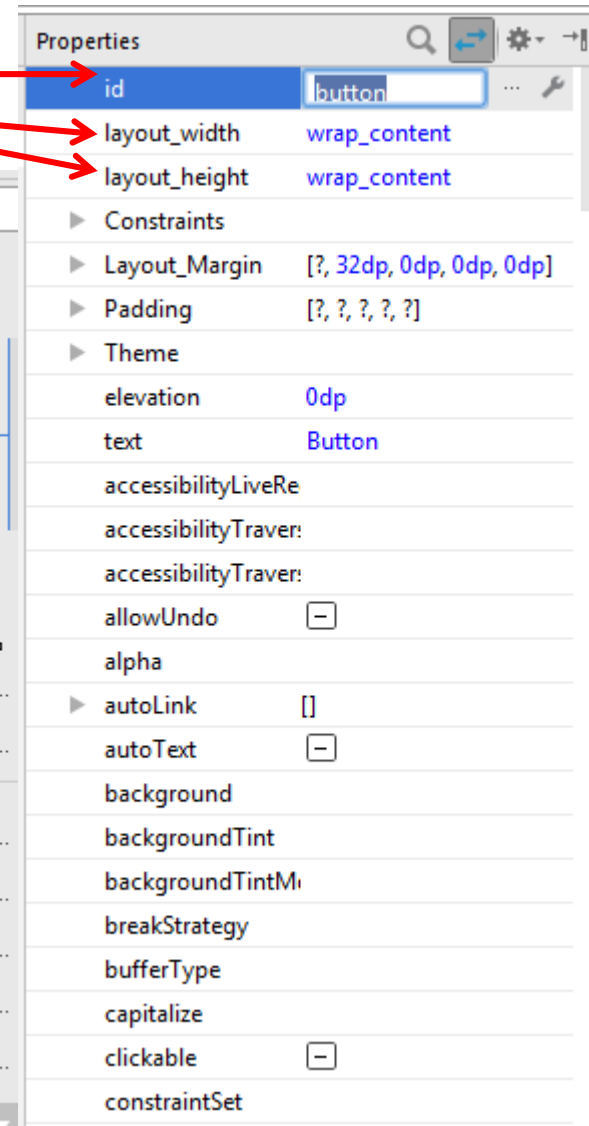
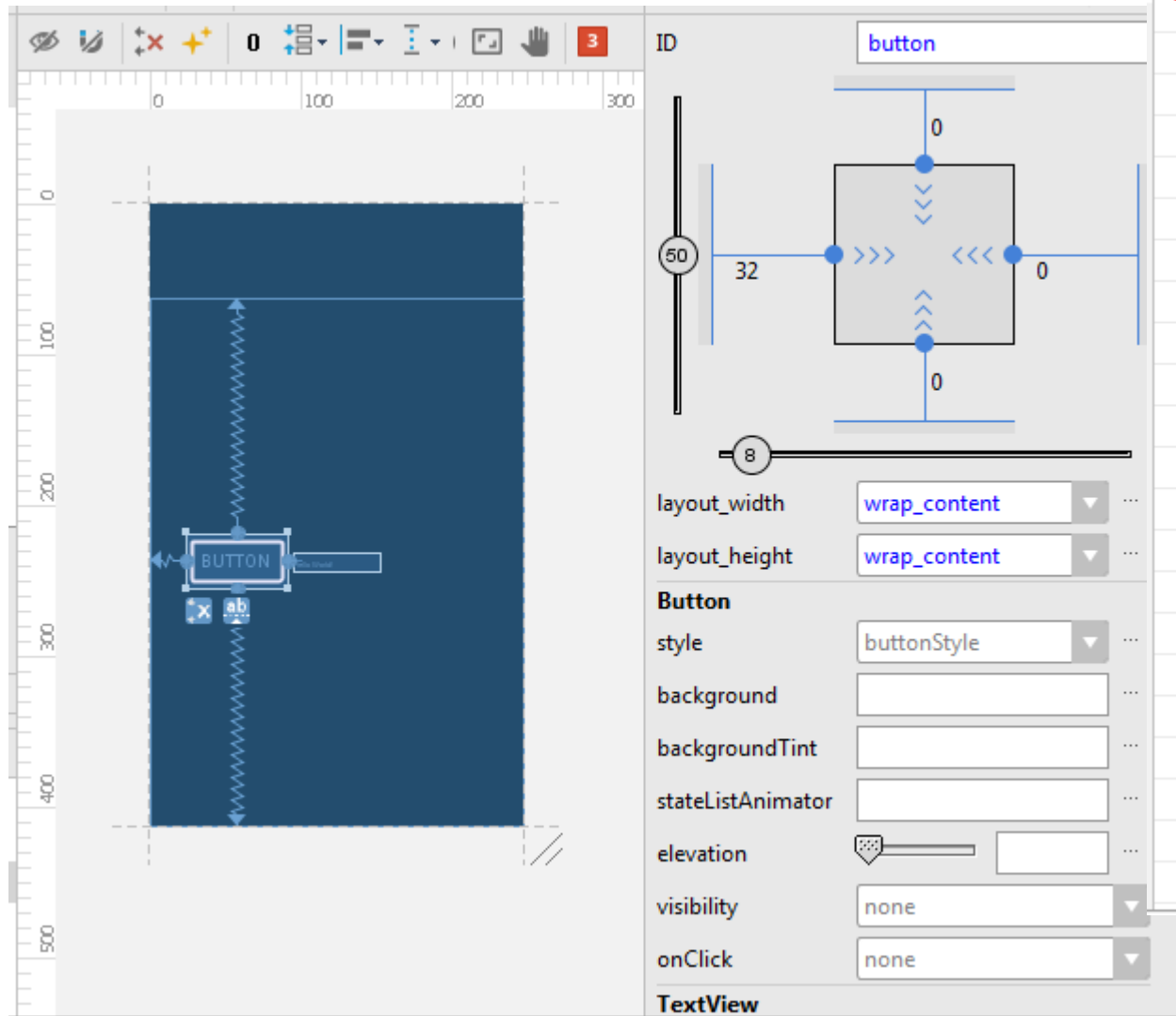


Задаем отступ



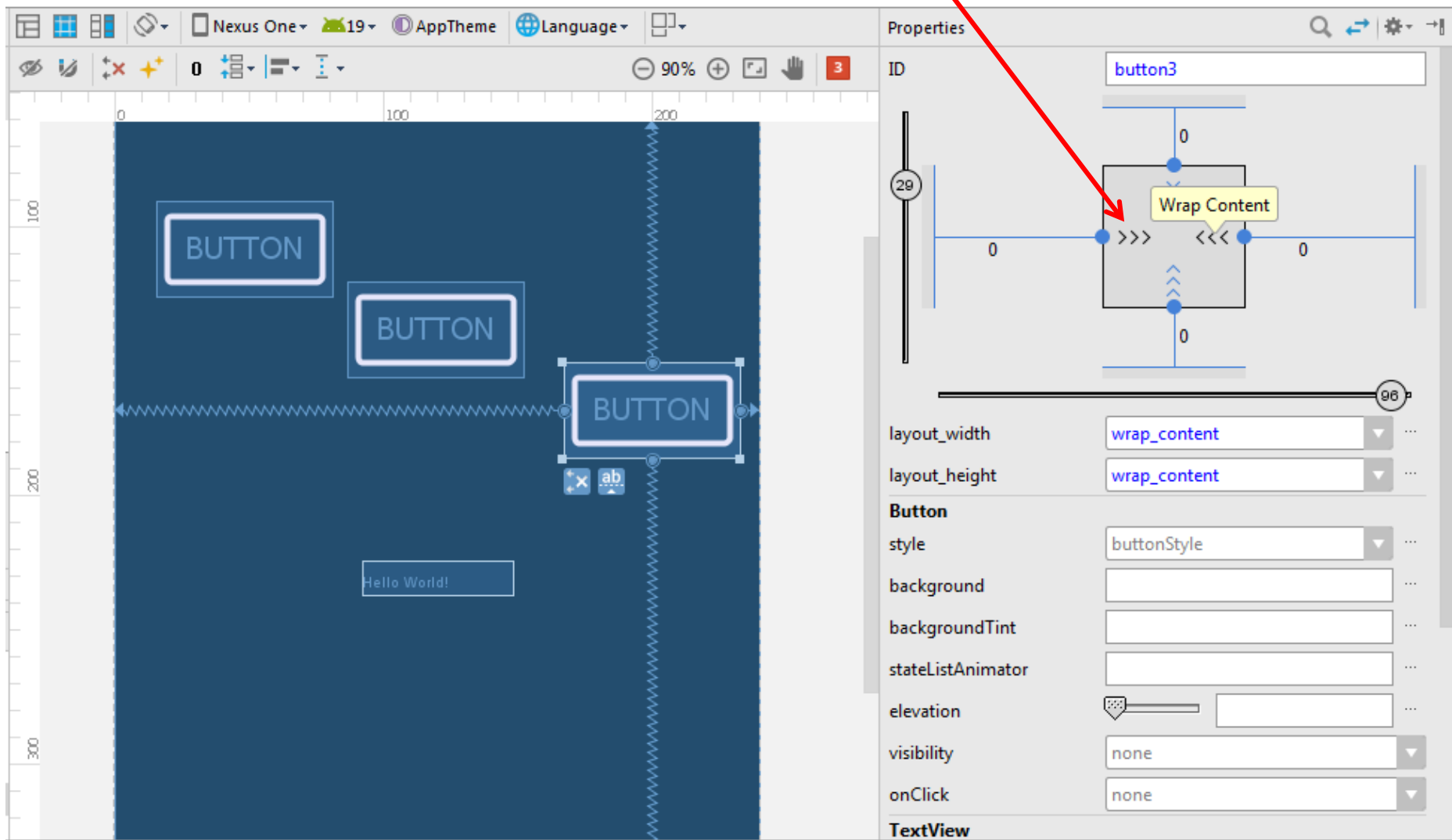
Виды Layout: ConstraintLayout

Меняем параметры: id, размеры



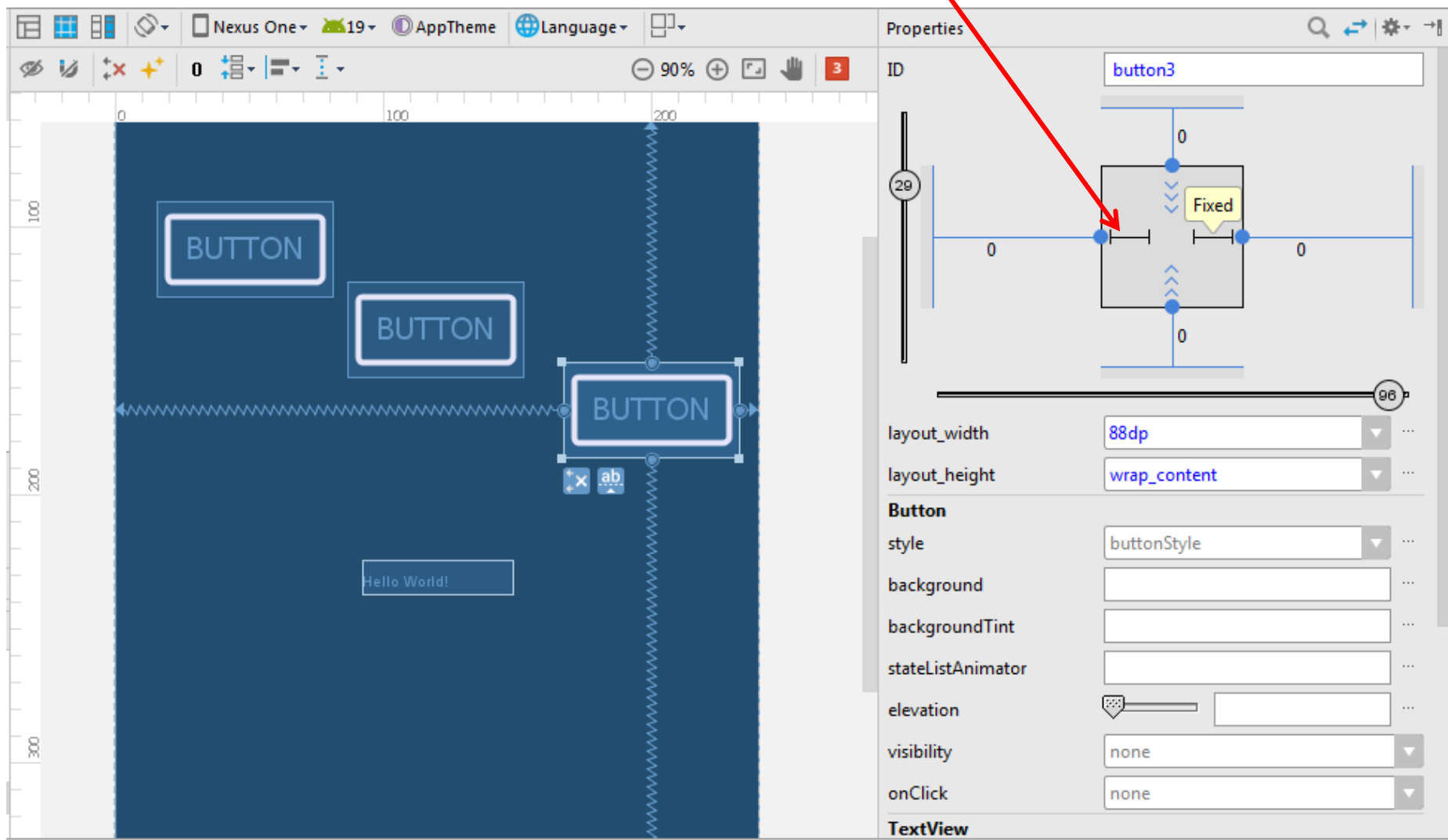
Виды Layout: ConstraintLayout

Три типа размеров: по содержимому



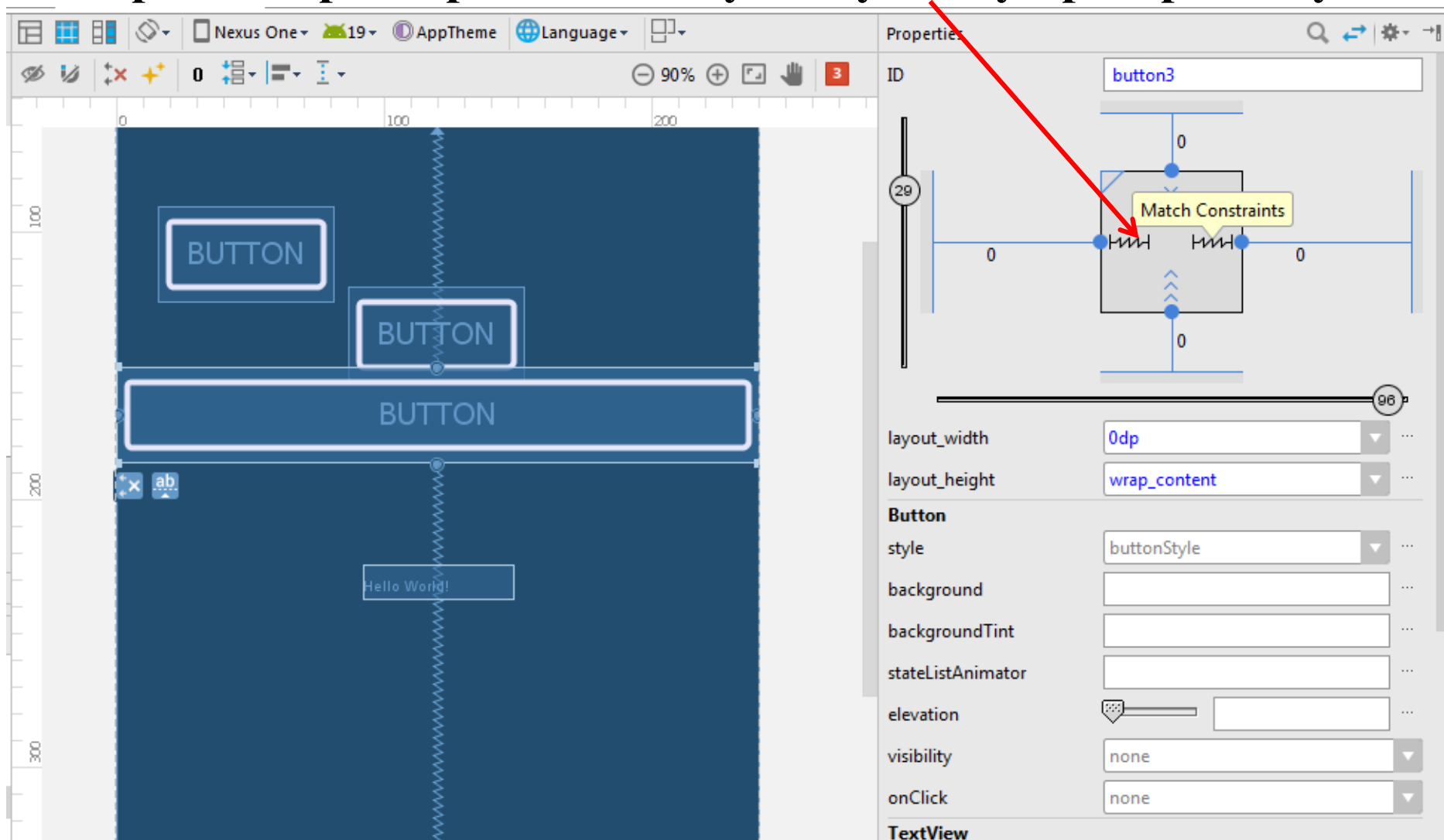
Виды Layout: ConstraintLayout

Три типа размеров: в dp (условных единицах)



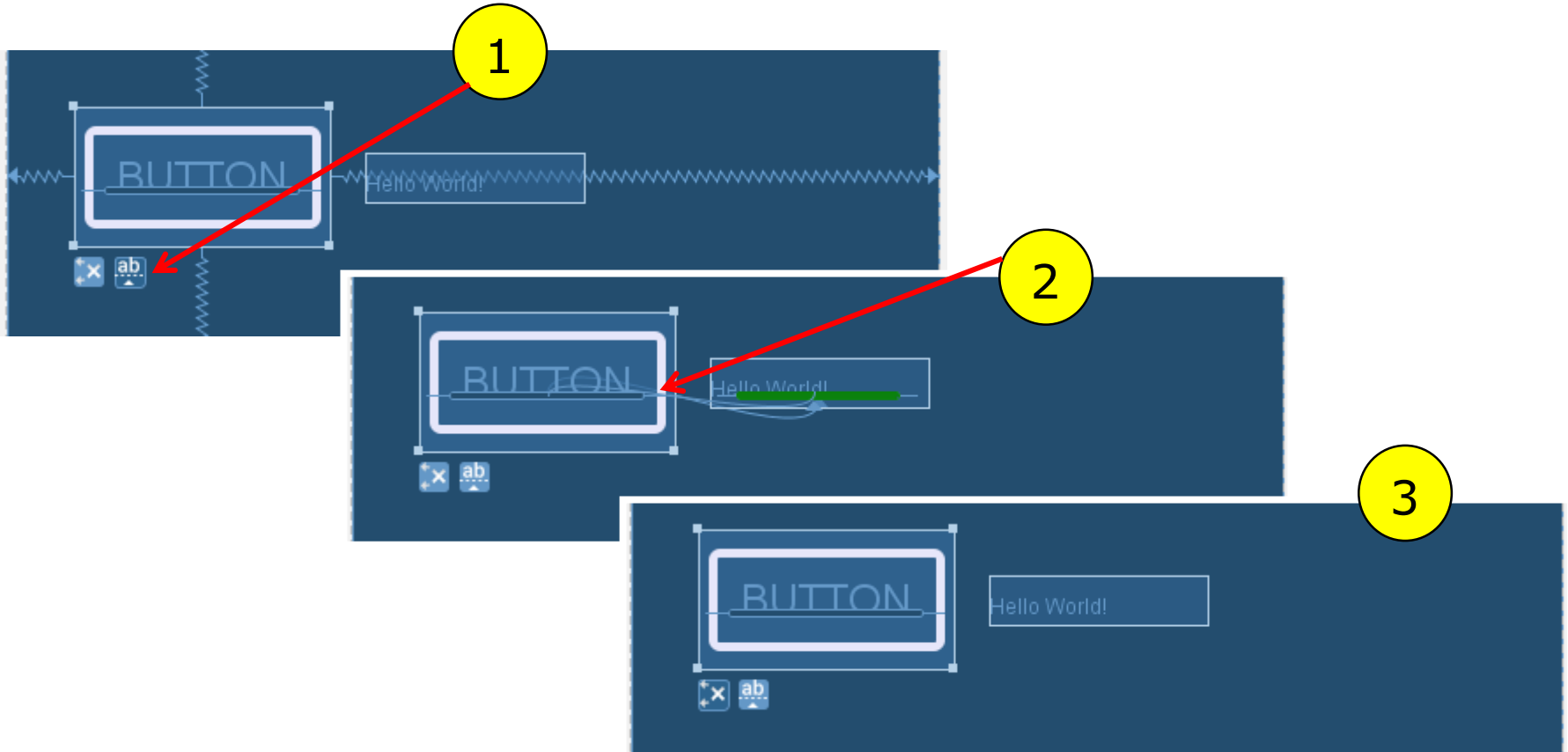
Виды Layout: ConstraintLayout

Три типа размеров: по всему доступному пространству



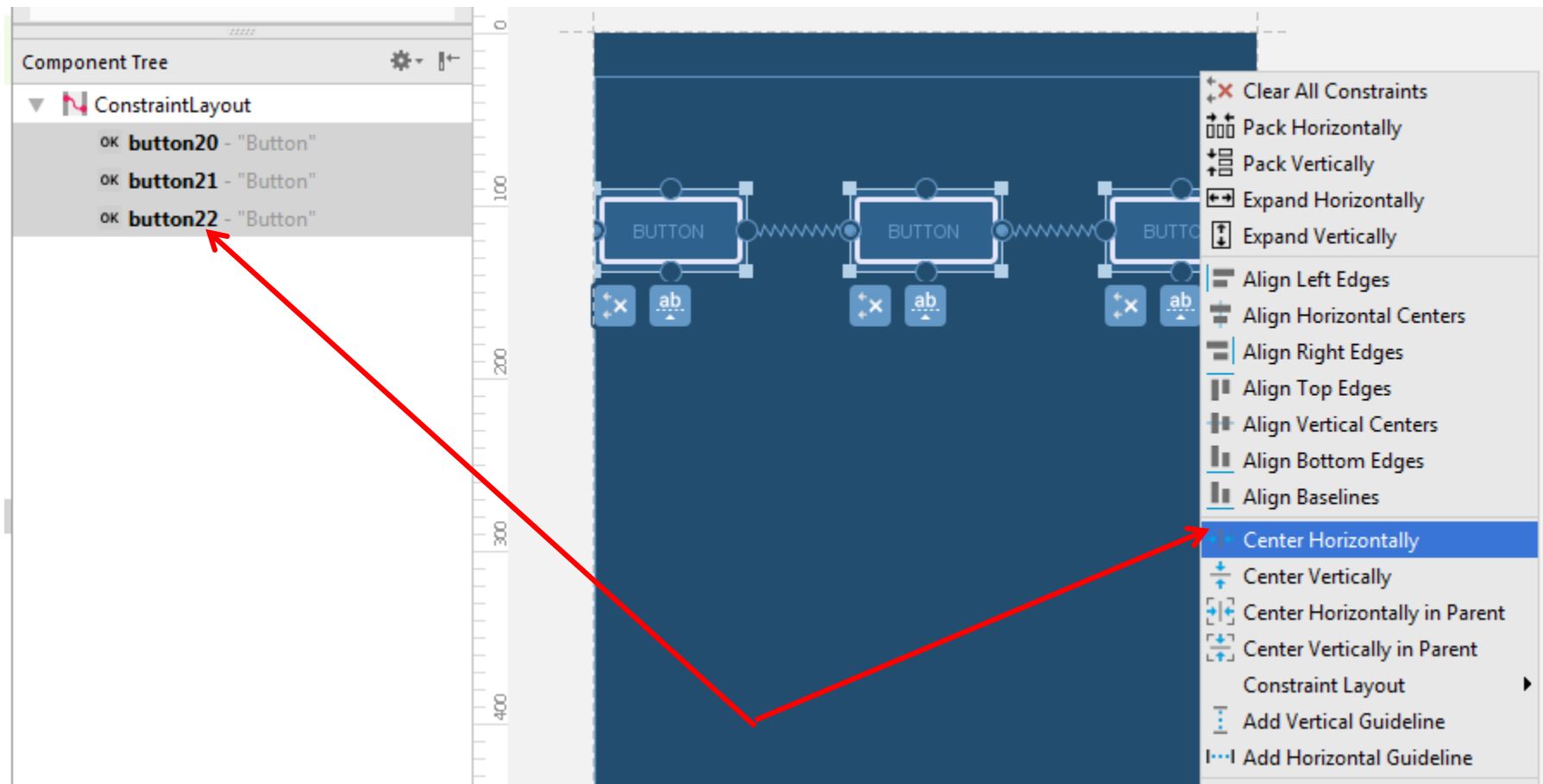
Виды Layout: ConstraintLayout

Выравниваем элементы по базовой линии



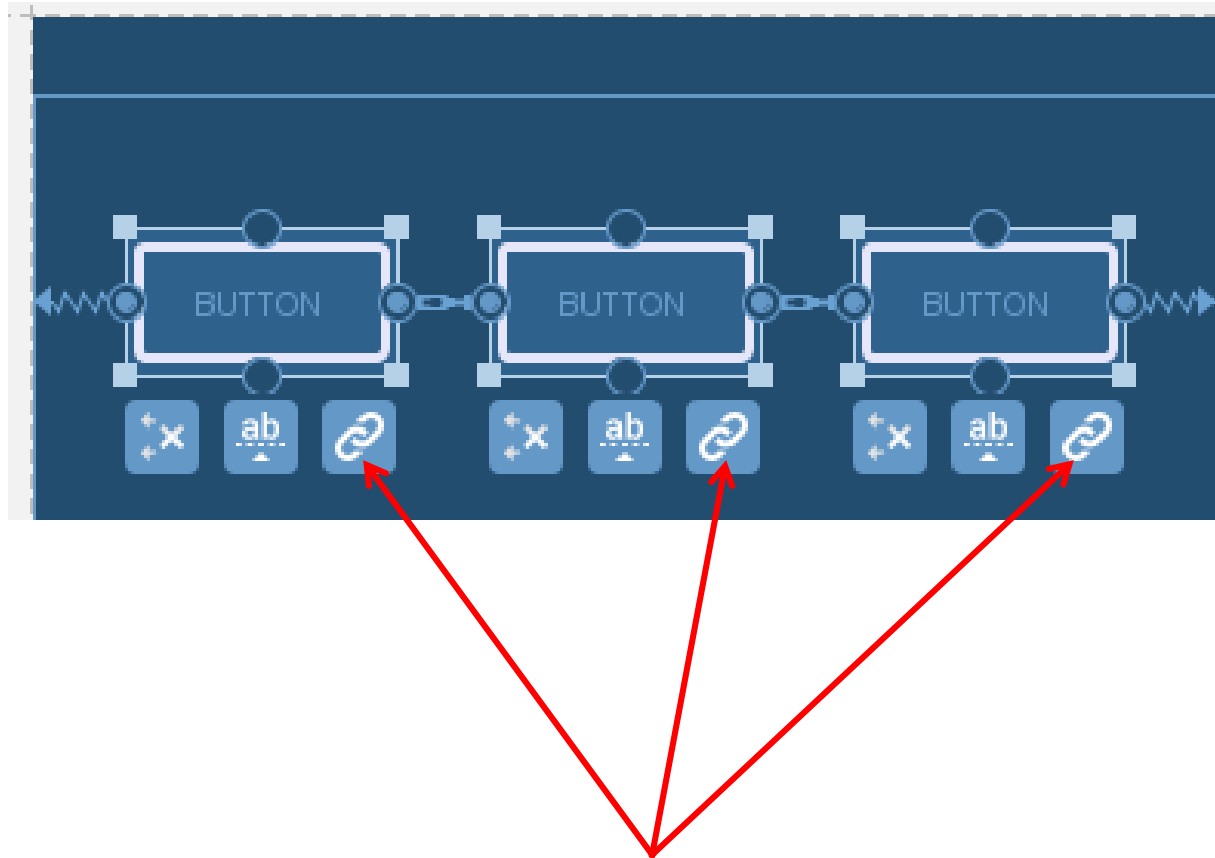
Виды Layout: ConstraintLayout

Объединяем элементы в единую цепь



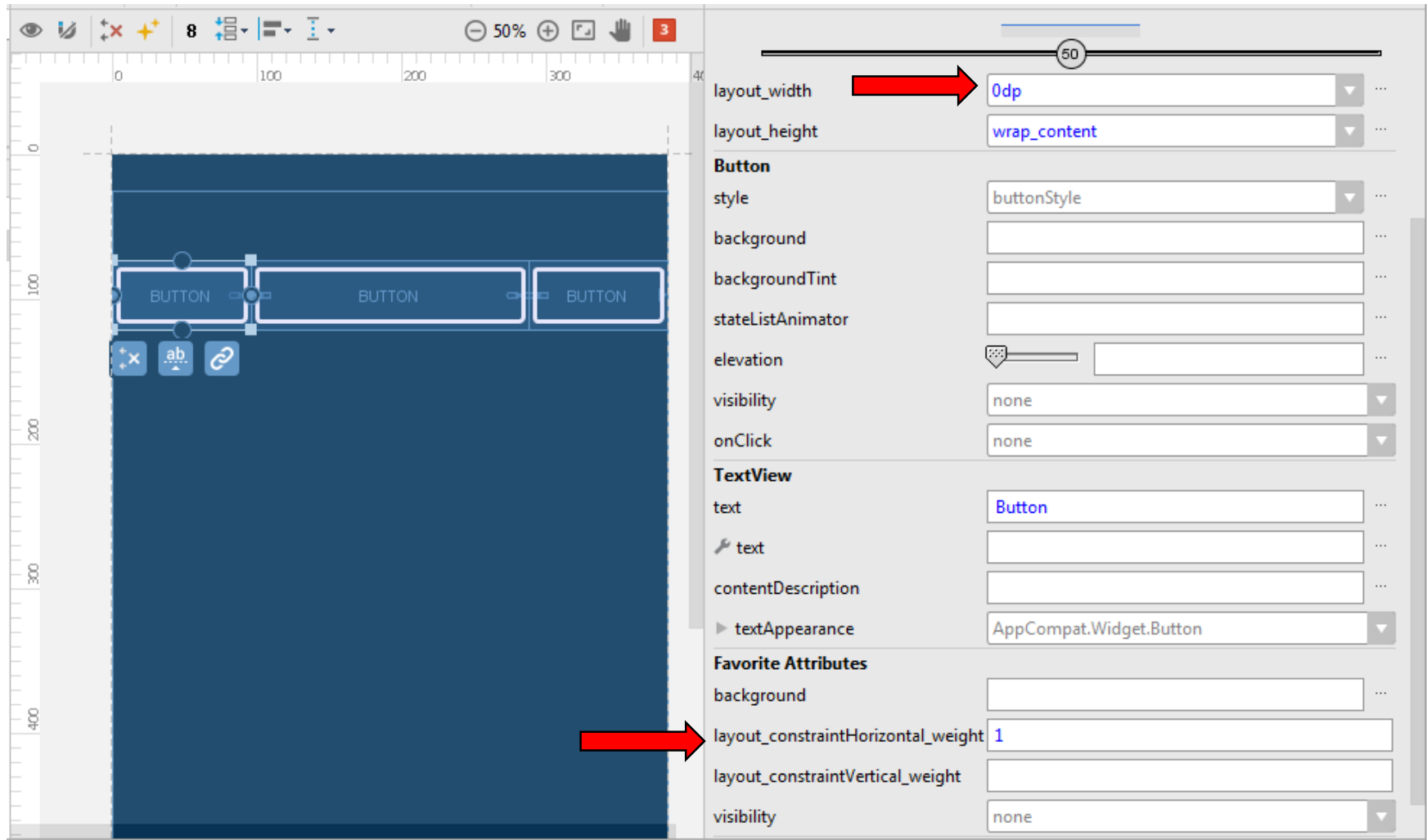
Виды Layout: ConstraintLayout

Объединяем элементы в единую цепь



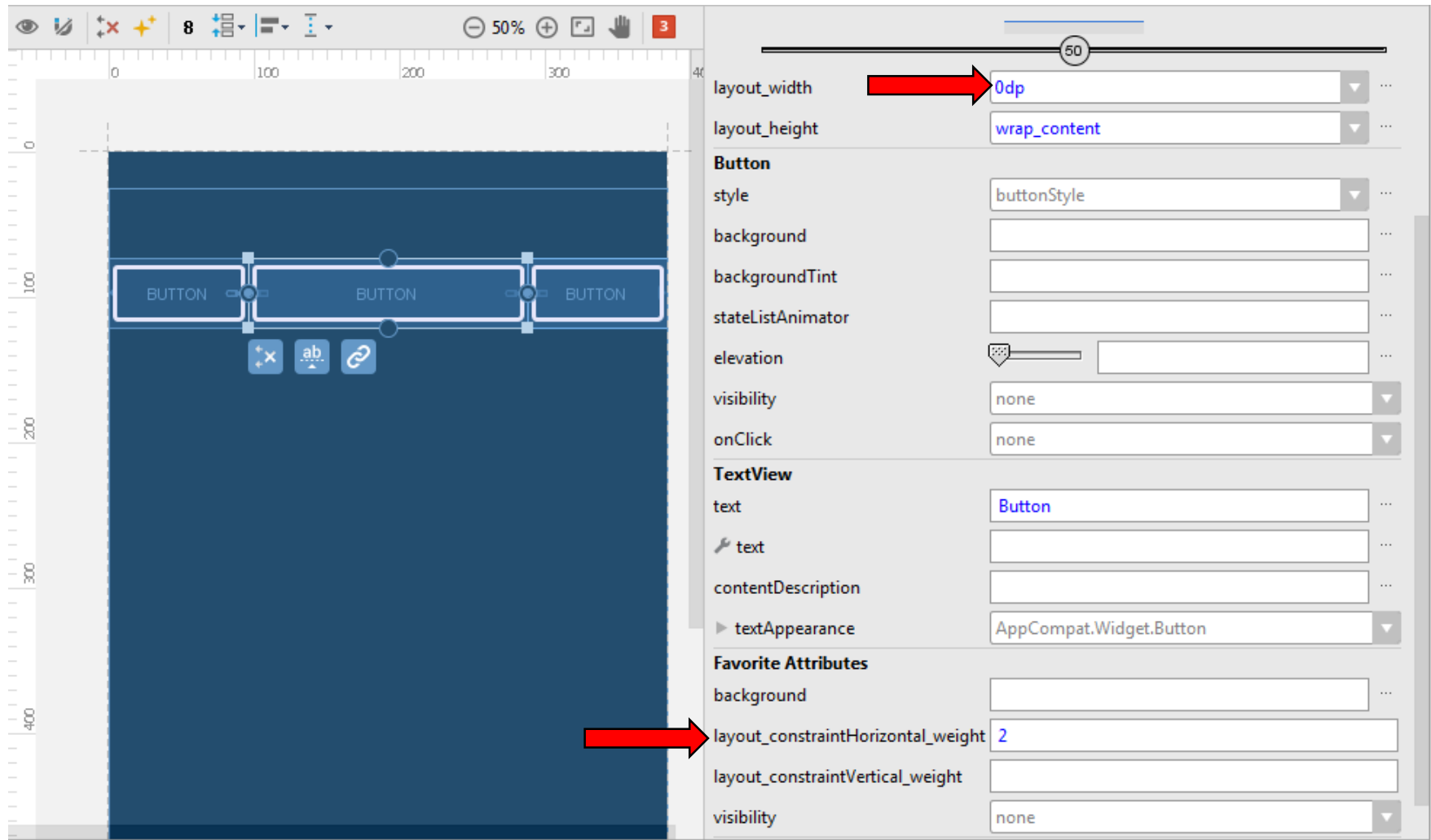
Виды Layout: ConstraintLayout

Задаем пропорциональные размеры



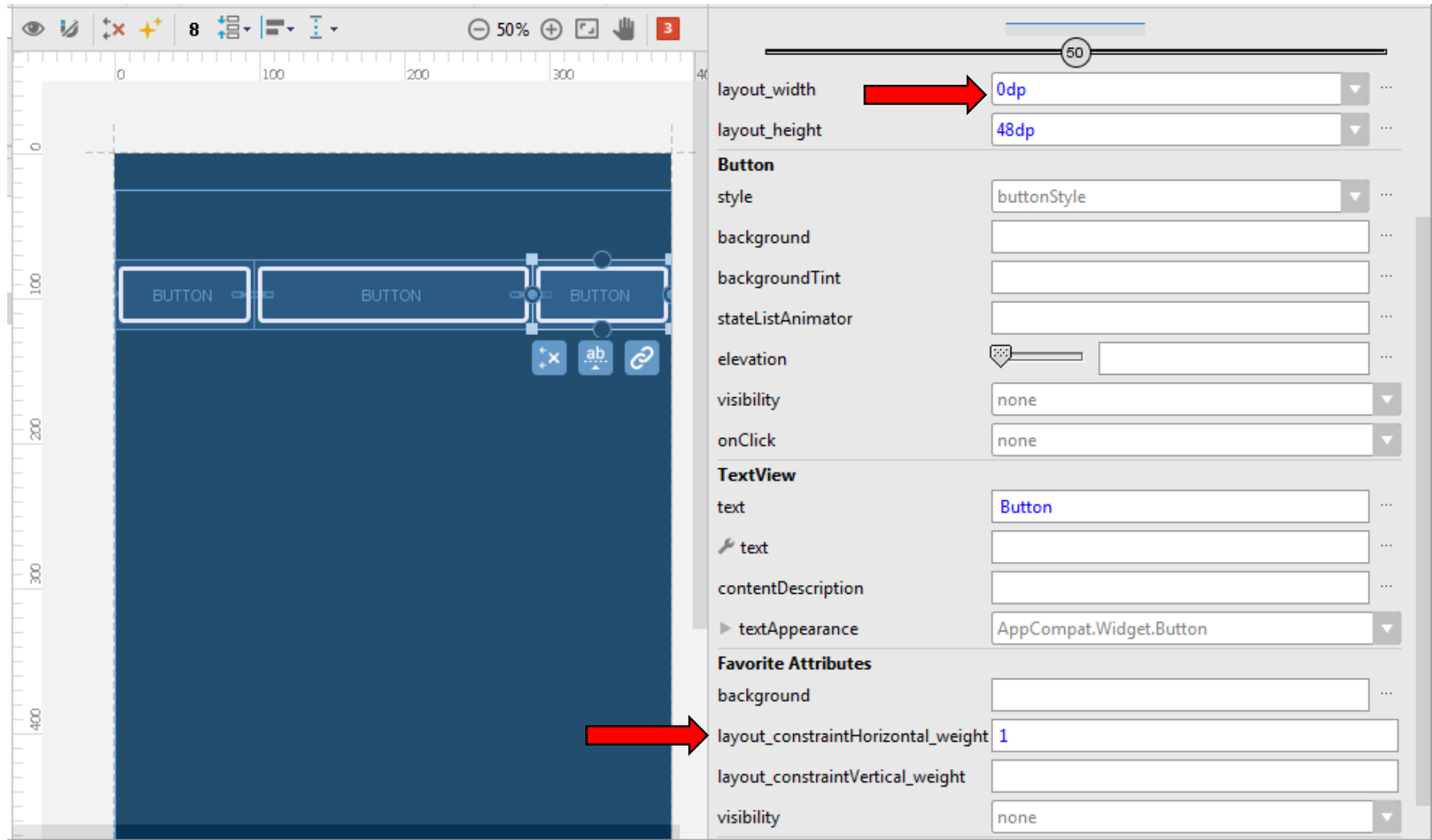
Виды Layout: ConstraintLayout

Задаем пропорциональные размеры



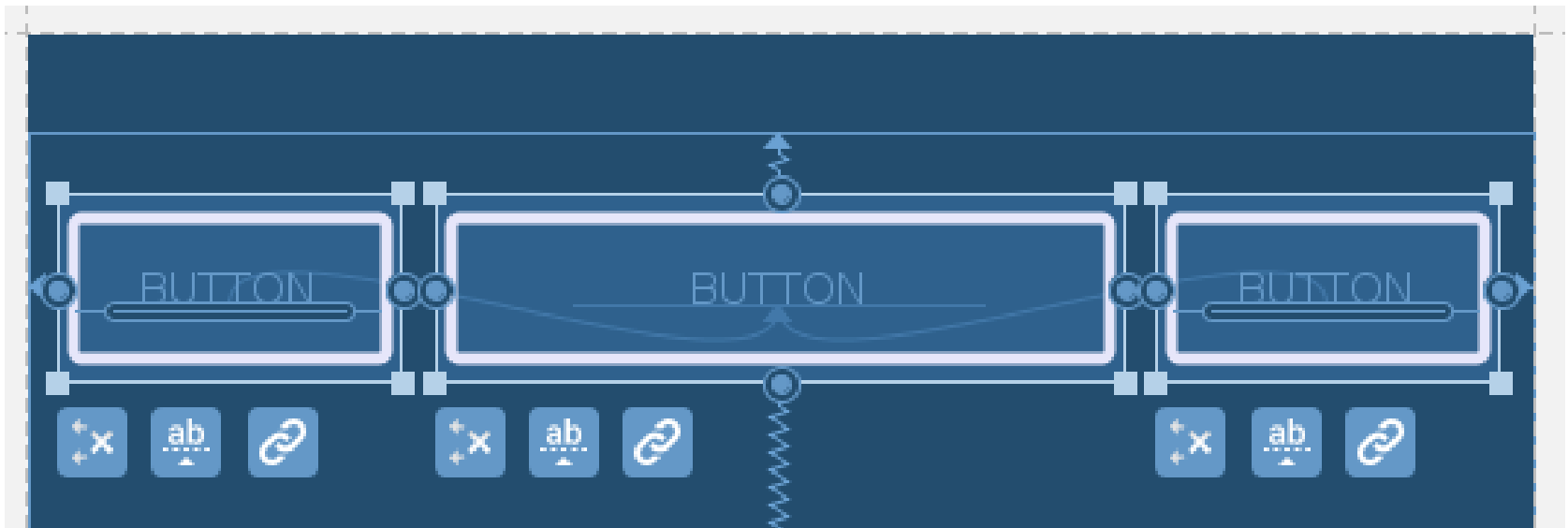
Виды Layout: ConstraintLayout

Задаем пропорциональные размеры



Виды Layout: ConstraintLayout

Добавляем выравнивание по базовой линии, отступы, вертикальную привязку



Параметры View-элементов

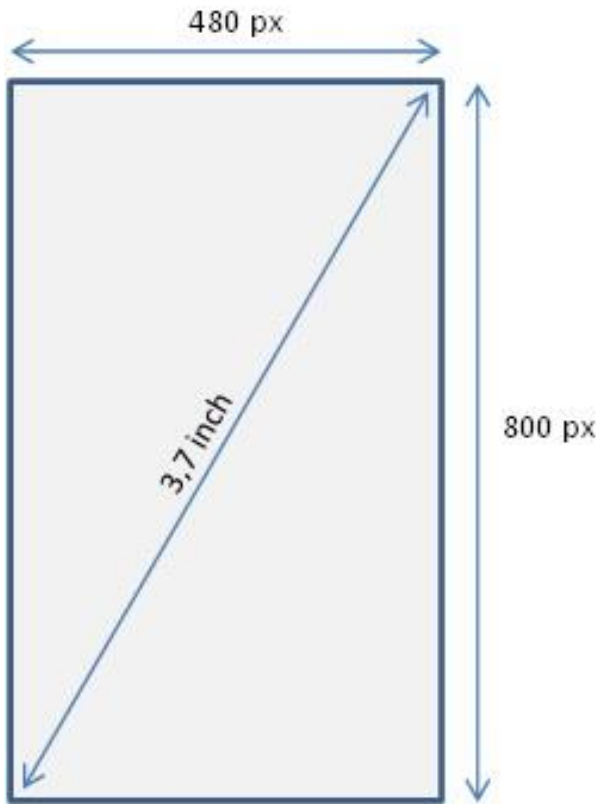
id – это ID элемента.

layout_width (ширина элемента) и **layout_height** (высота элемента) могут задаваться в абстрактных значениях (**dp**), а могут быть следующими:

- **fill_parent** или **match_parent** (максимально возможная ширина или высота в пределах родителя),
- **wrap_content** (ширина или высота определяется по содержимому элемента).

```
<Button  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Button 5"  
    android:id="@+id/button5" />
```

Параметры View-элементов



dp - Density-independent Pixels. Абстрактная ЕИ, позволяющая приложениям выглядеть одинаково на различных экранах и разрешениях.

sp - Scale-independent Pixels. То же, что и dp, только используется для размеров шрифта в View элементах

pt - 1/72 дюйма, определяется по физическому размеру экрана.

px — пиксел, не рекомендуется использовать т.к. на разных экранах приложение будет выглядеть по-разному.

mm — миллиметр, определяется по физическому размеру экрана.

in — дюйм, определяется по физическому размеру экрана.

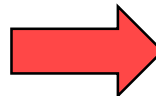
Параметры View-элементов

Layout weight – вес – делит пространство между элементами.

```
<Button  
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:text="B1"  
    android:id="@+id/button1"  
    android:layout_weight="1"/>
```

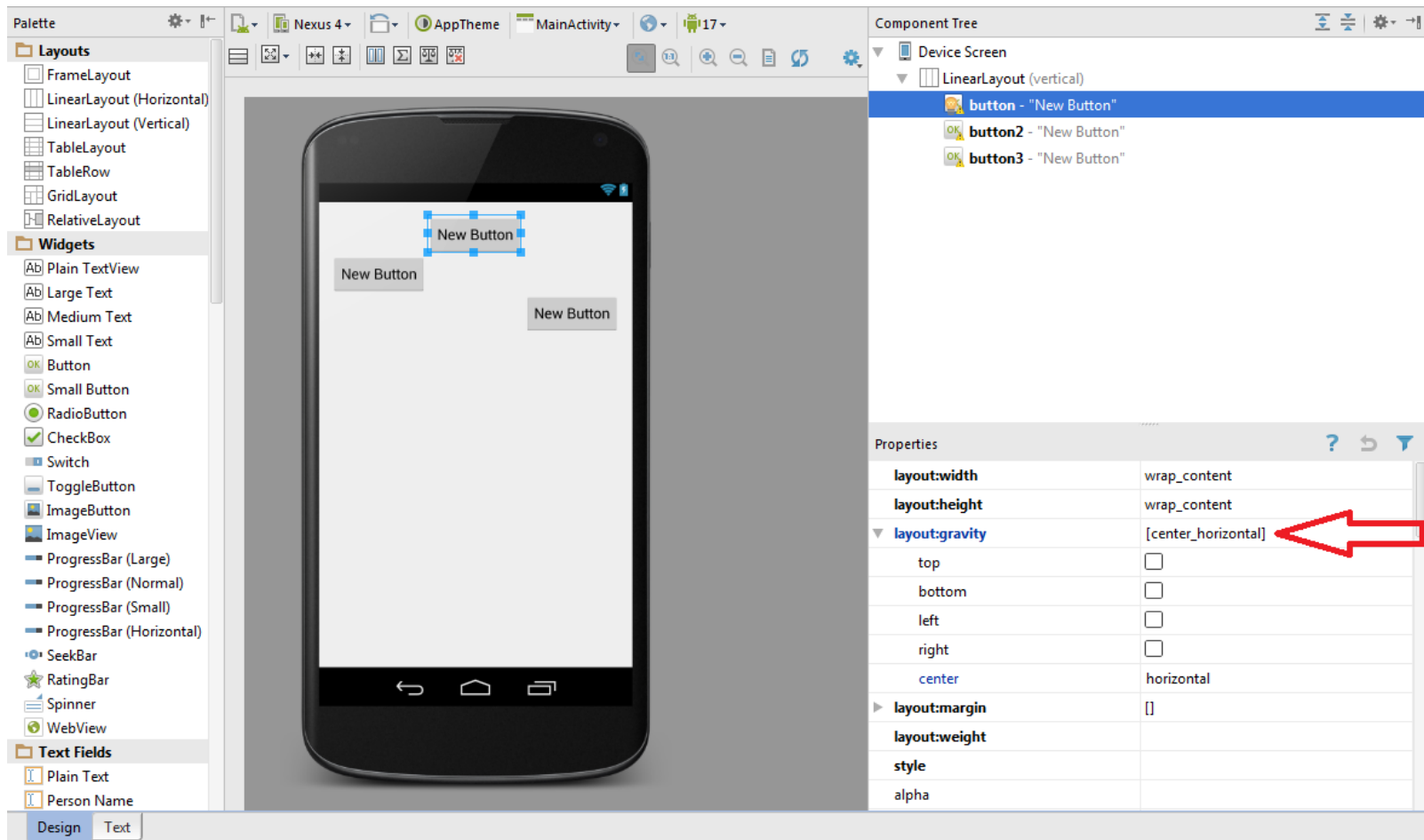
```
<Button  
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:text="B2"  
    android:id="@+id/button2"  
    android:layout_weight="2"/>
```

```
<Button  
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:text="B3"  
    android:id="@+id/button3"  
    android:layout_weight="3"/>
```



Параметры View-элементов

Layout gravity – выравнивание.



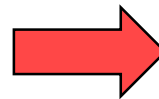
Параметры View-элементов

Layout gravity – выравнивание.

```
<Button  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="New Button"  
    android:id="@+id/button"  
    android:layout_gravity="center horizontal" />
```

```
<Button  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="New Button"  
    android:id="@+id/button2" />
```

```
<Button  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="New Button"  
    android:id="@+id/button3"  
    android:layout_gravity="right" />
```



Параметры View-элементов

Layout margin – отступ.

The screenshot illustrates the Android Studio interface for designing a mobile application. The central workspace shows a virtual smartphone screen with three buttons, each labeled "New Button". The left sidebar contains the "Layouts" and "Widgets" toolbars. The right sidebar displays the "Device Screen" hierarchy, showing a "LinearLayout (vertical)" containing three buttons: "button - 'New Button'", "button2 - 'New Button'", and "button3 - 'New Button'". The "Properties" panel at the bottom right shows the "layout:margin" property set to "[10dp, ?, ?, ?, ?, ?]", with a red arrow pointing to it.

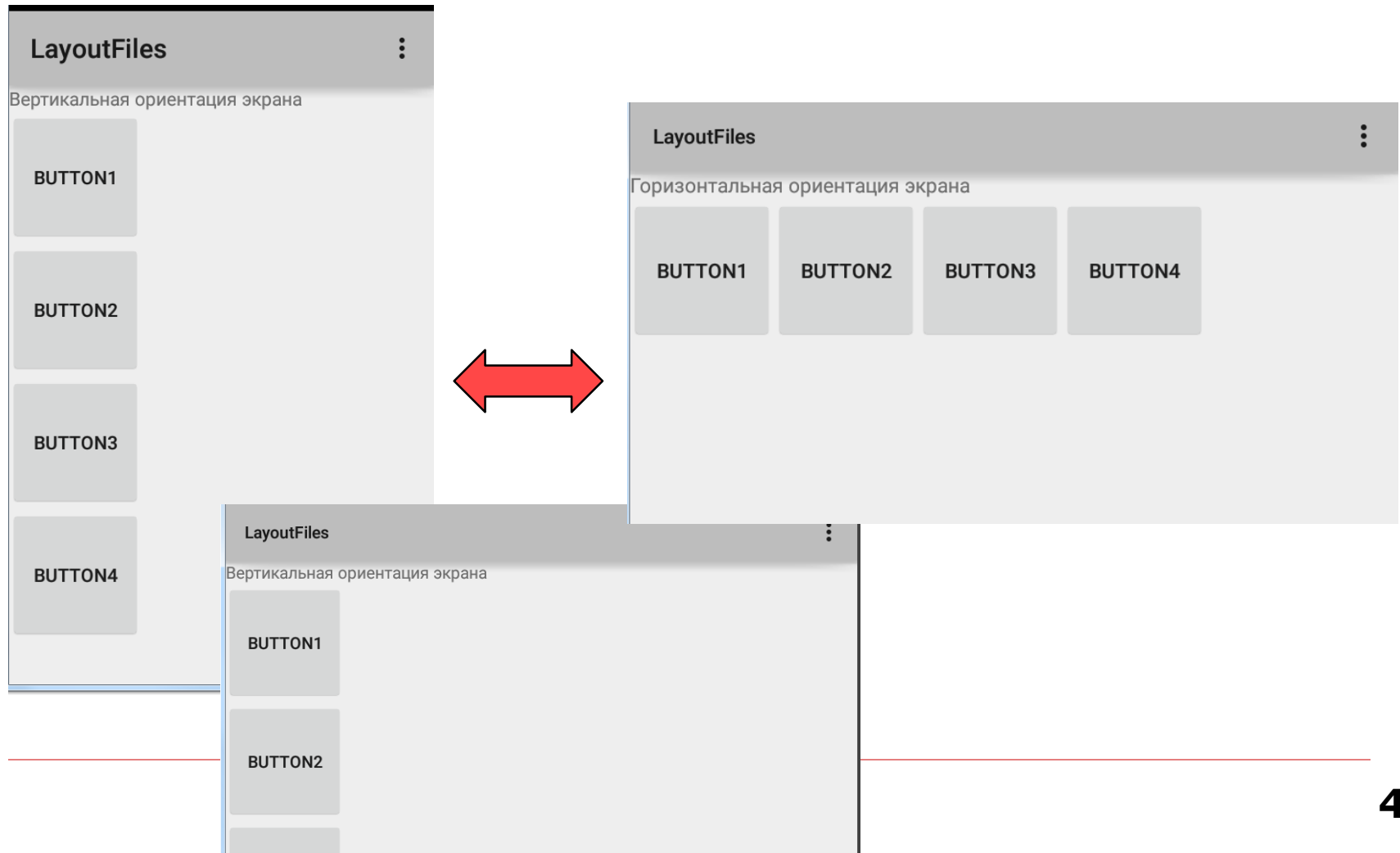
| Property | Value |
|----------------|-----------------------|
| layout:width | wrap_content |
| layout:height | wrap_content |
| layout:gravity | [center_horizontal] |
| layout:margin | [10dp, ?, ?, ?, ?, ?] |
| all | 10dp |
| left | |
| top | |
| right | |
| bottom | |
| layout:weight | |
| style | |
| alpha | |
| background | |
| capitalize | |

Параметры View-элементов

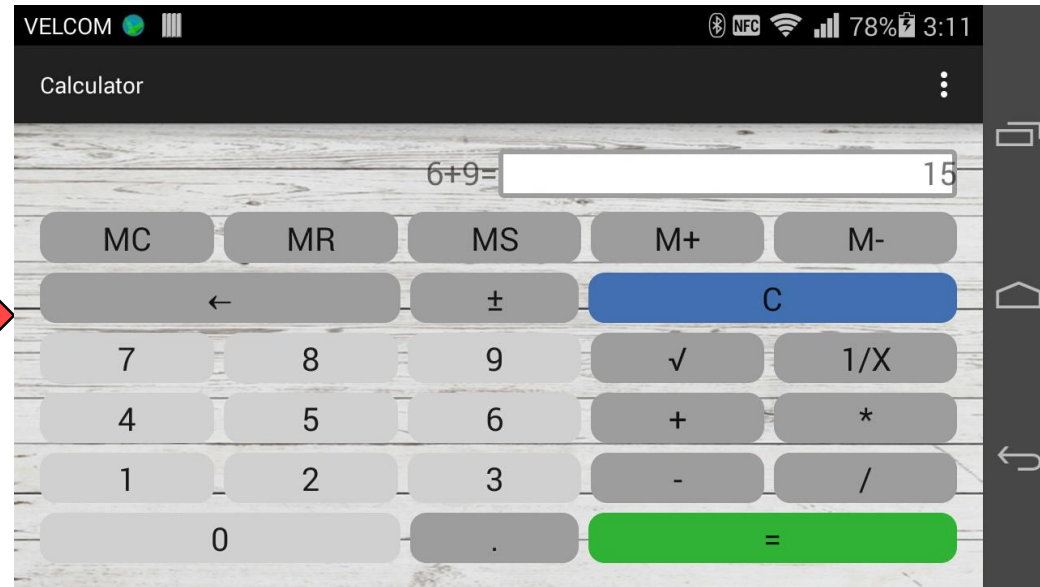
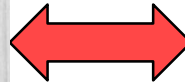
Layout margin – отступ.



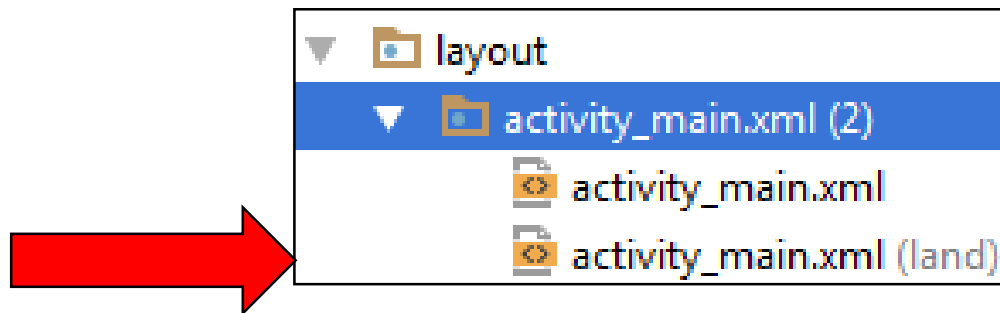
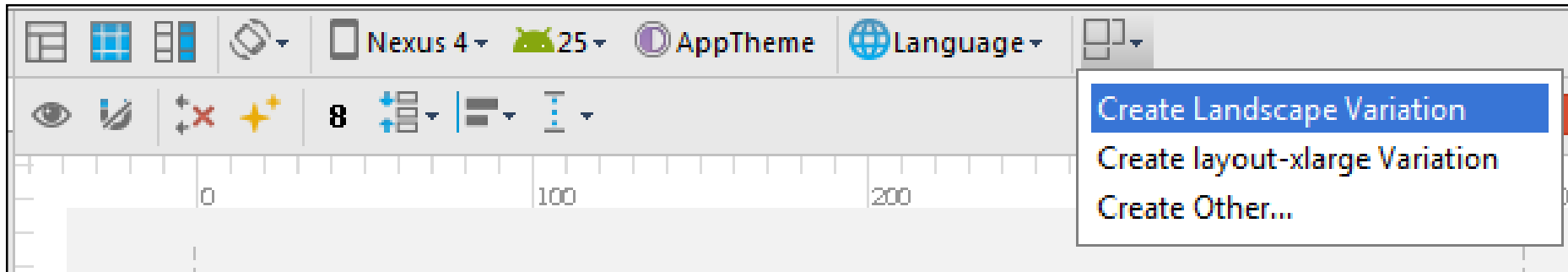
Activity как основа Android-приложения: смена ориентации экрана



Activity как основа Android-приложения: смена ориентации экрана

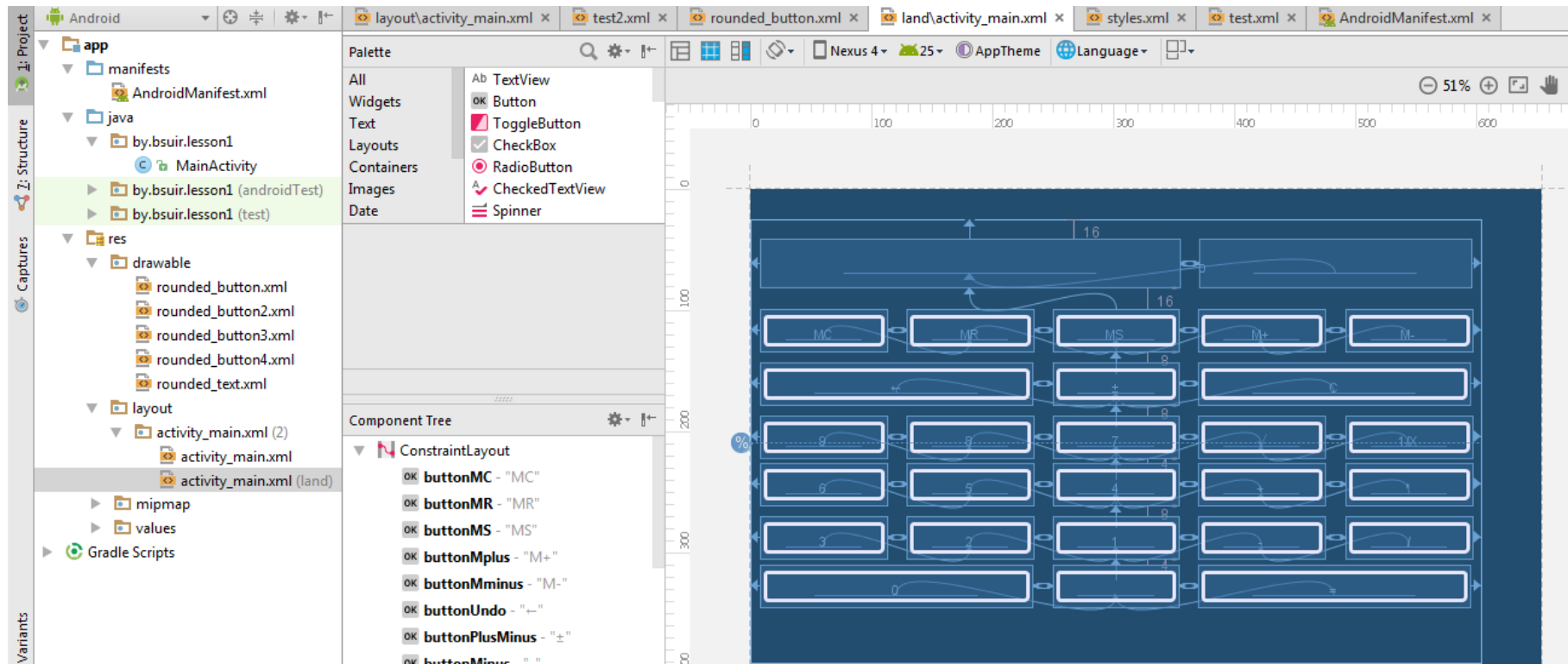


Activity как основа Android-приложения: смена ориентации экрана



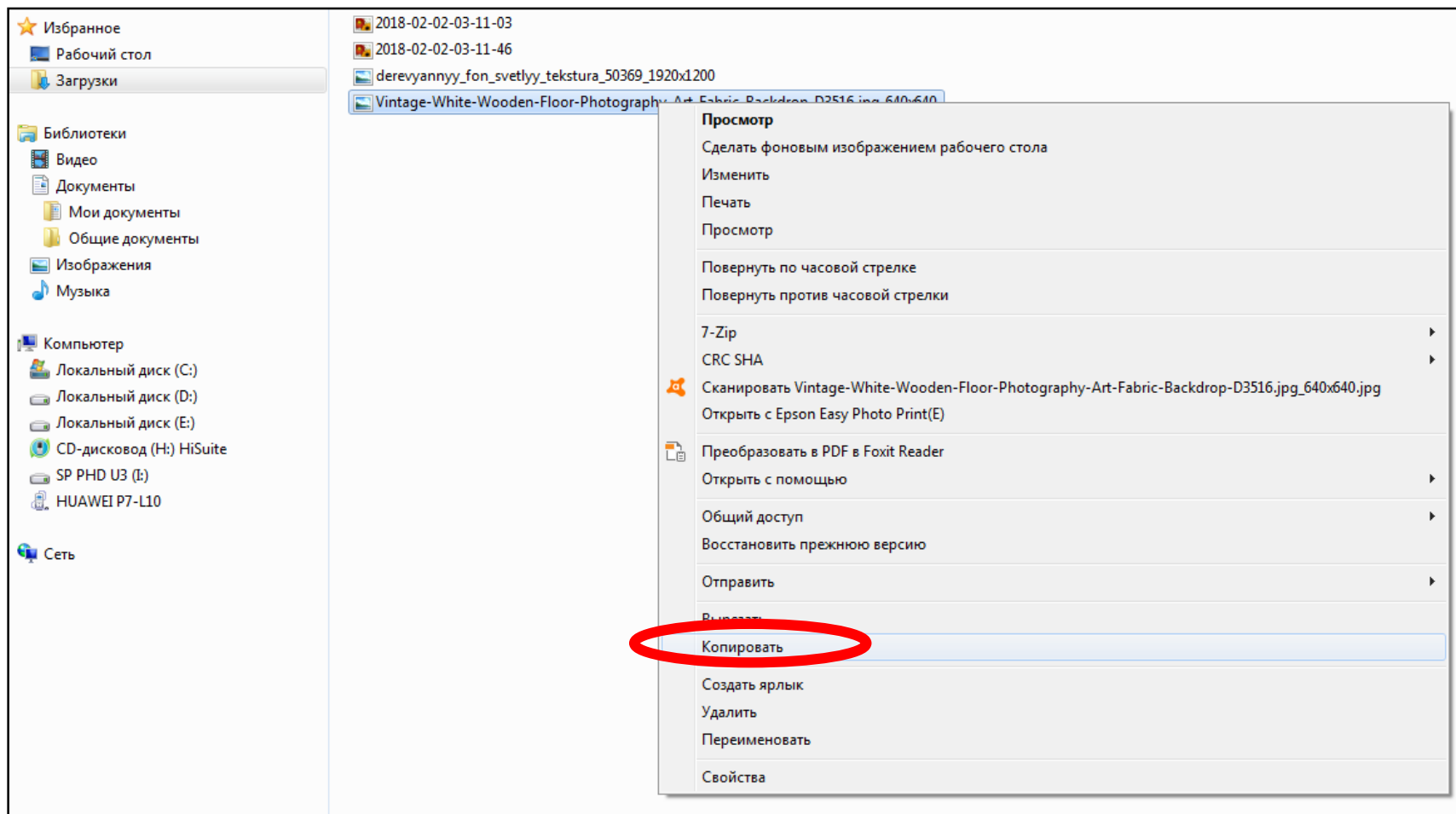
Activity как основа Android-приложения: смена ориентации экрана

! Используйте элементы с теми же ID



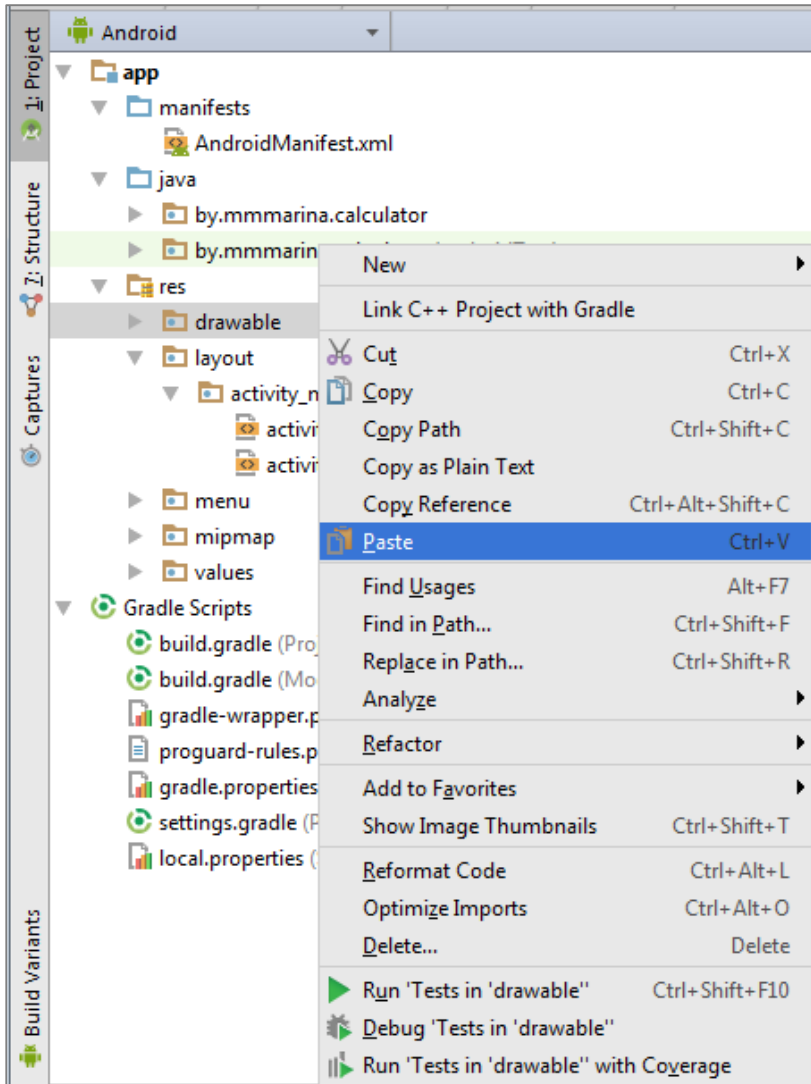
Добавляем фоновое изображение

1. Копируем изображение:



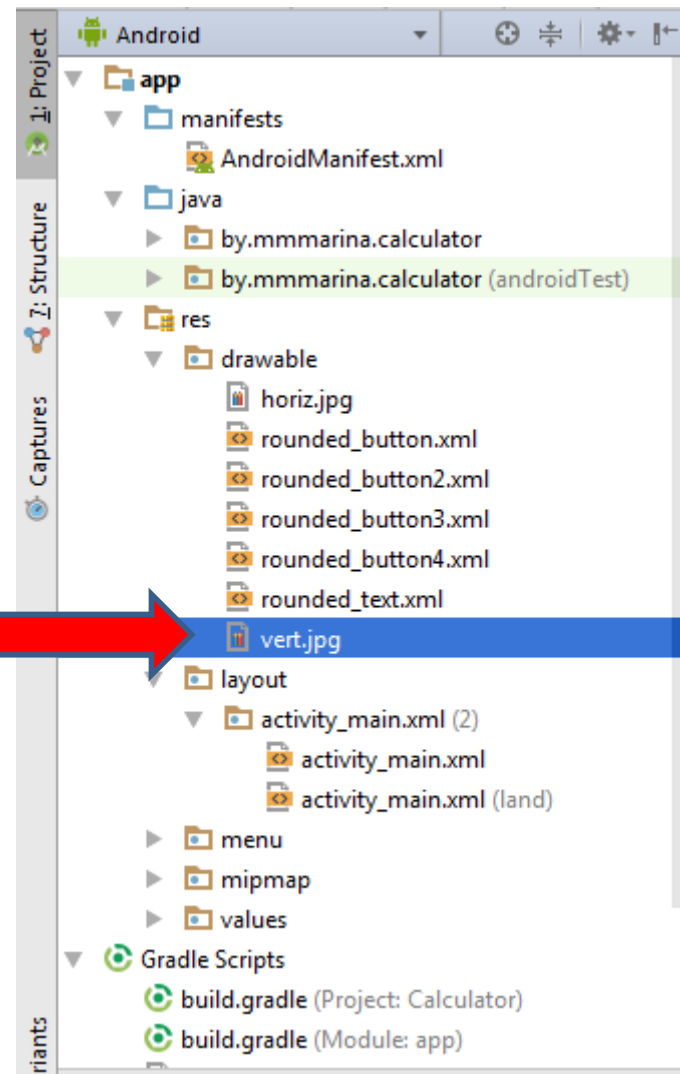
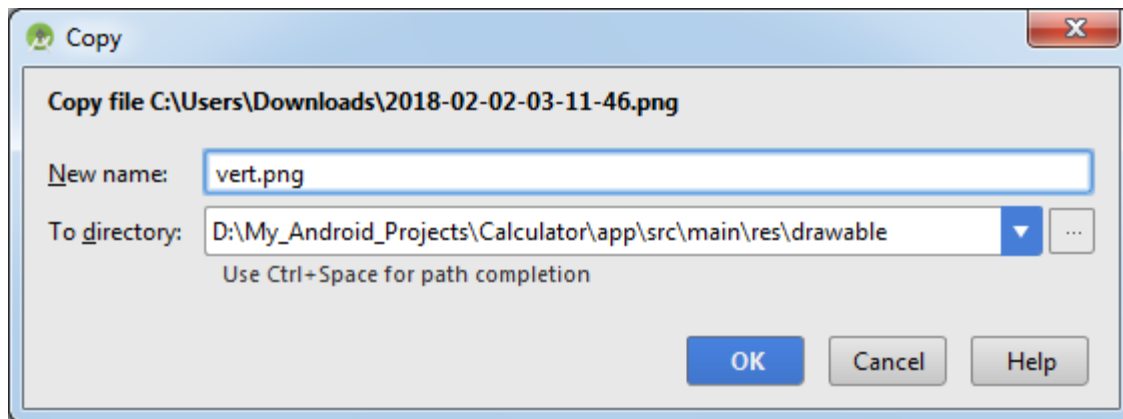
Добавляем фоновое изображение

2. Вставляем в папку drawable:



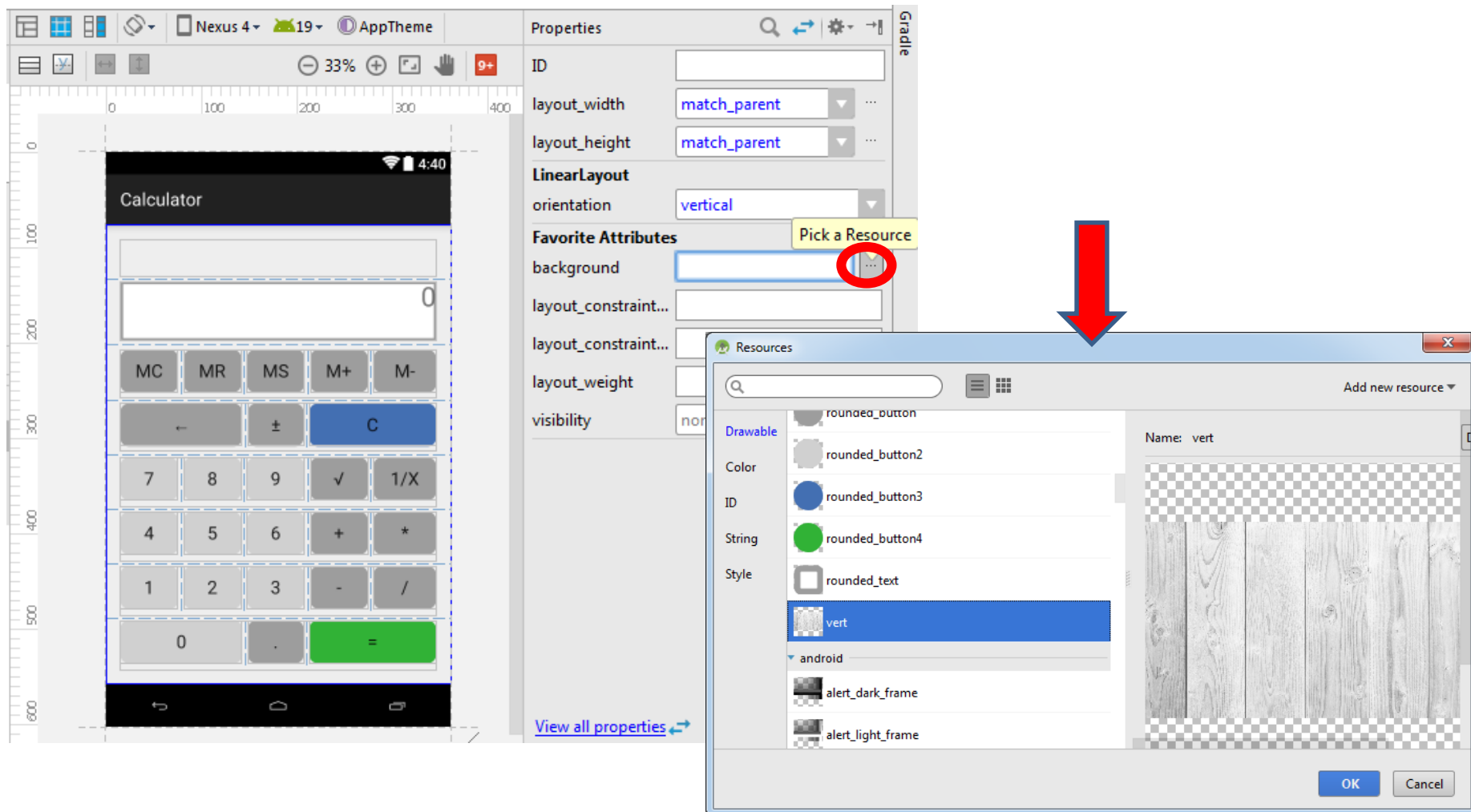
Добавляем фоновое изображение

3. Даем осмысленное имя:



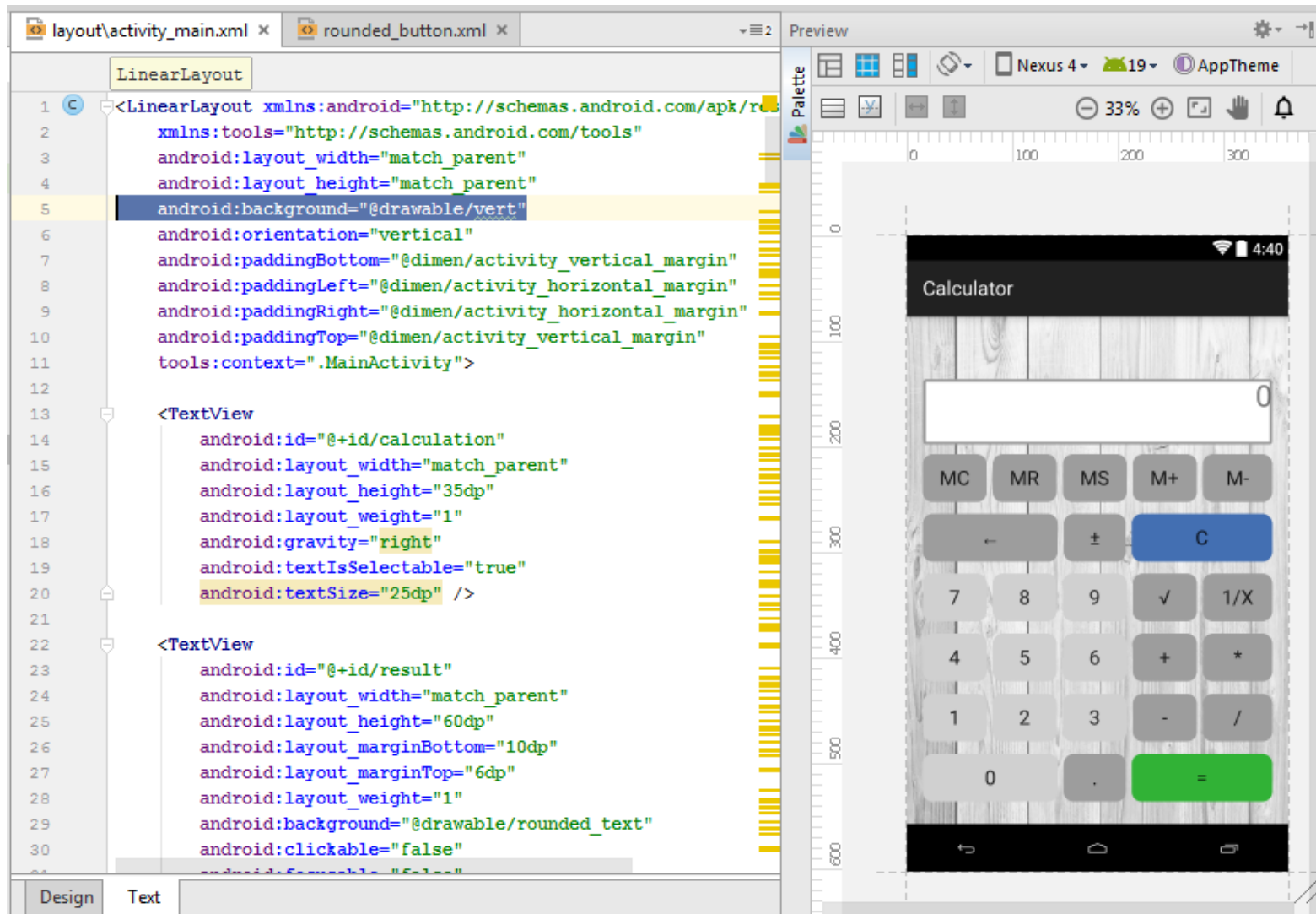
Добавляем фоновое изображение

4. Подключаем изображение к корневому layout:



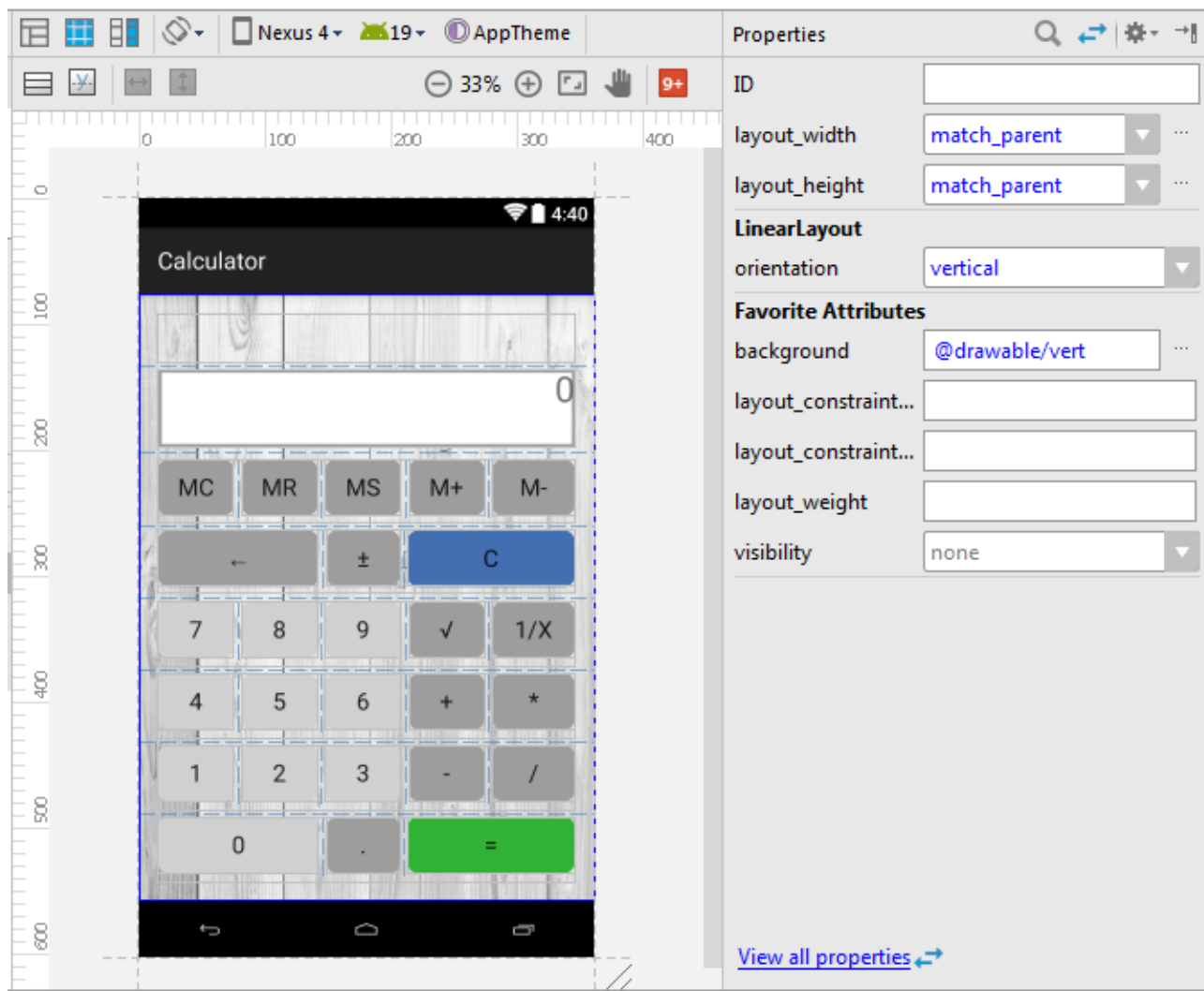
Добавляем фоновое изображение

4. Подключаем изображение к корневому layout (способ 2, через xml):



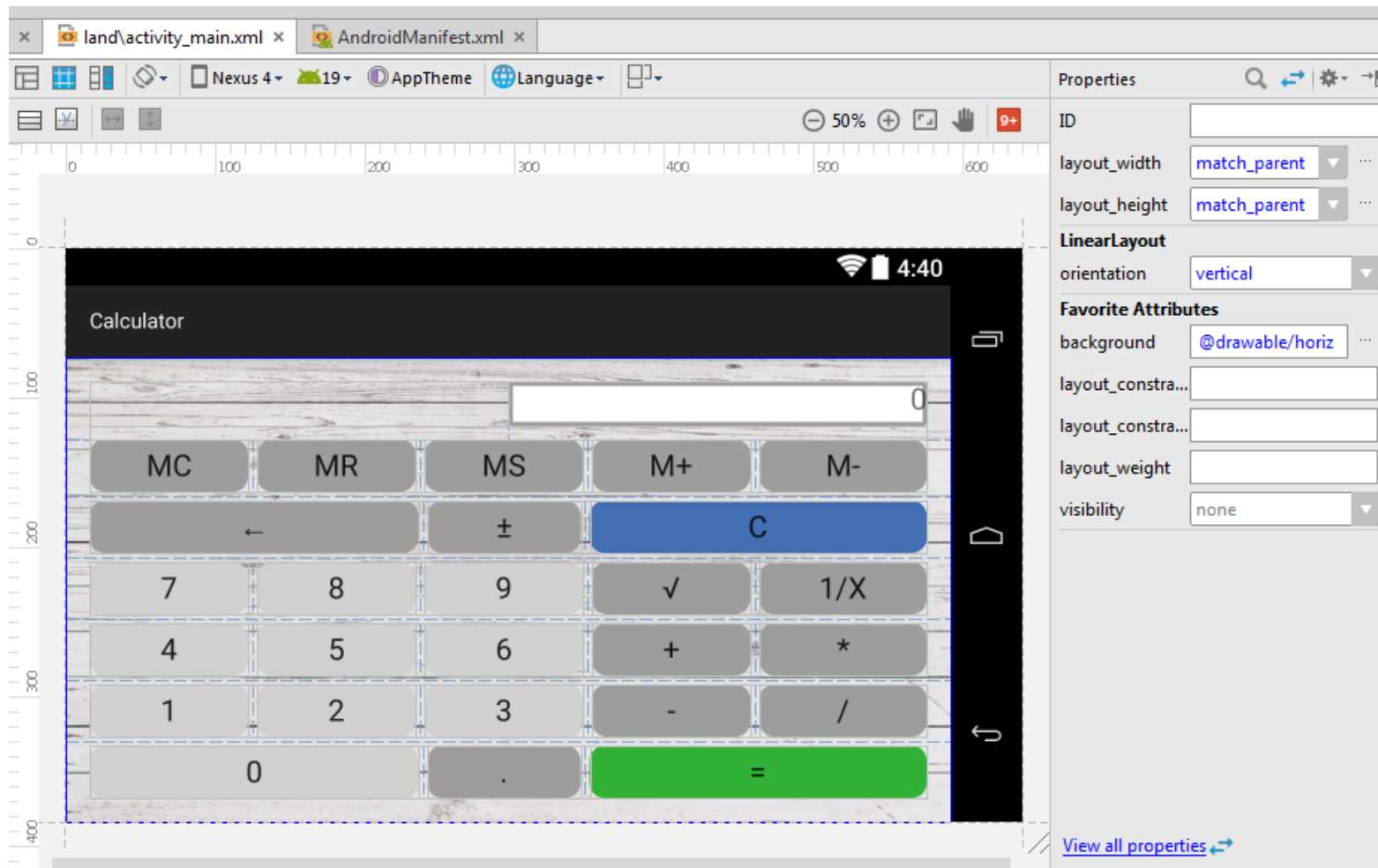
Добавляем фоновое изображение

Результат:



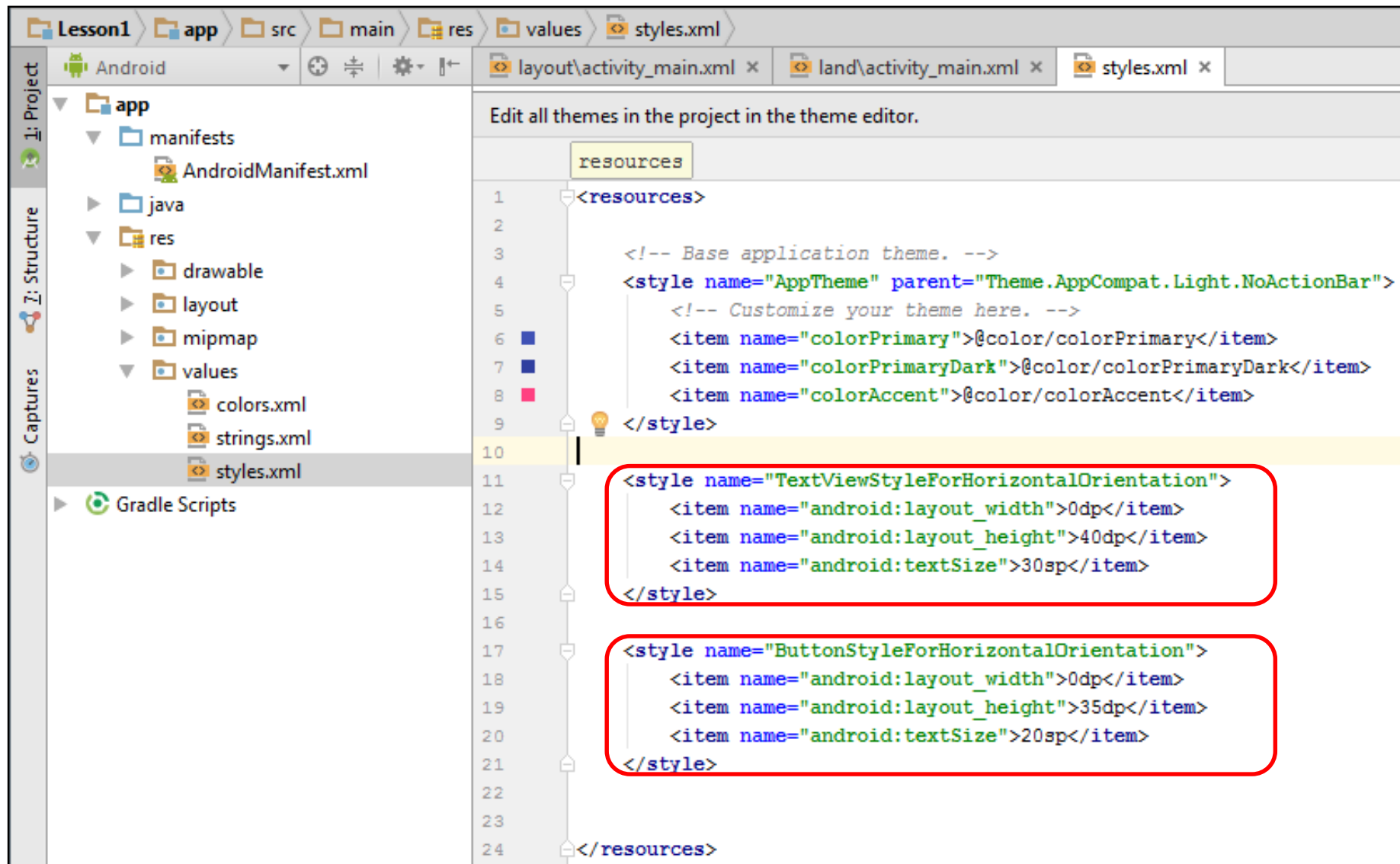
Добавляем фоновое изображение

Продолжаем в том же порядке:



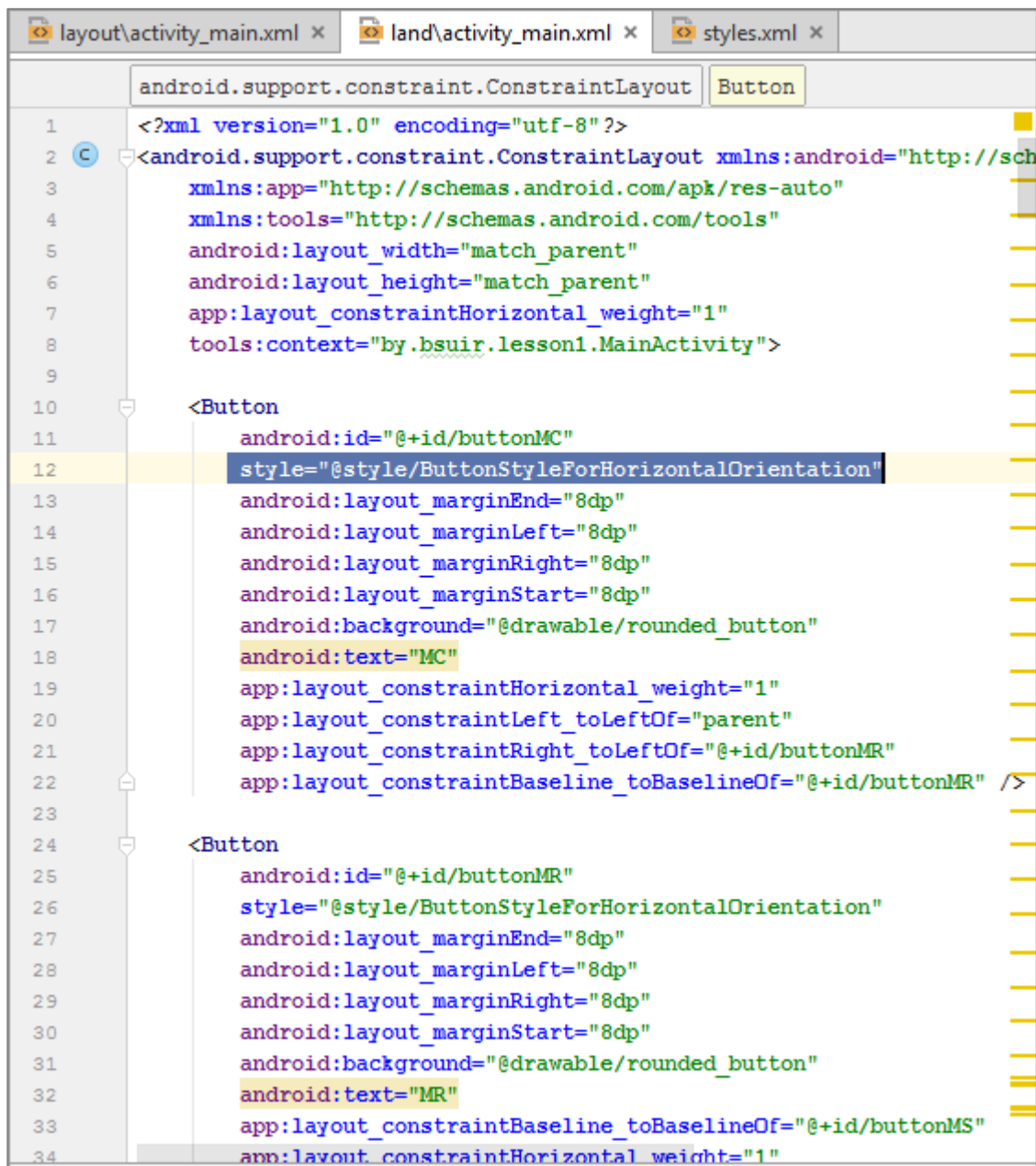
Создаем свои стили

Создаем в styles.xml свои стили с общими для компонентов свойствами:



Создаем свои стили

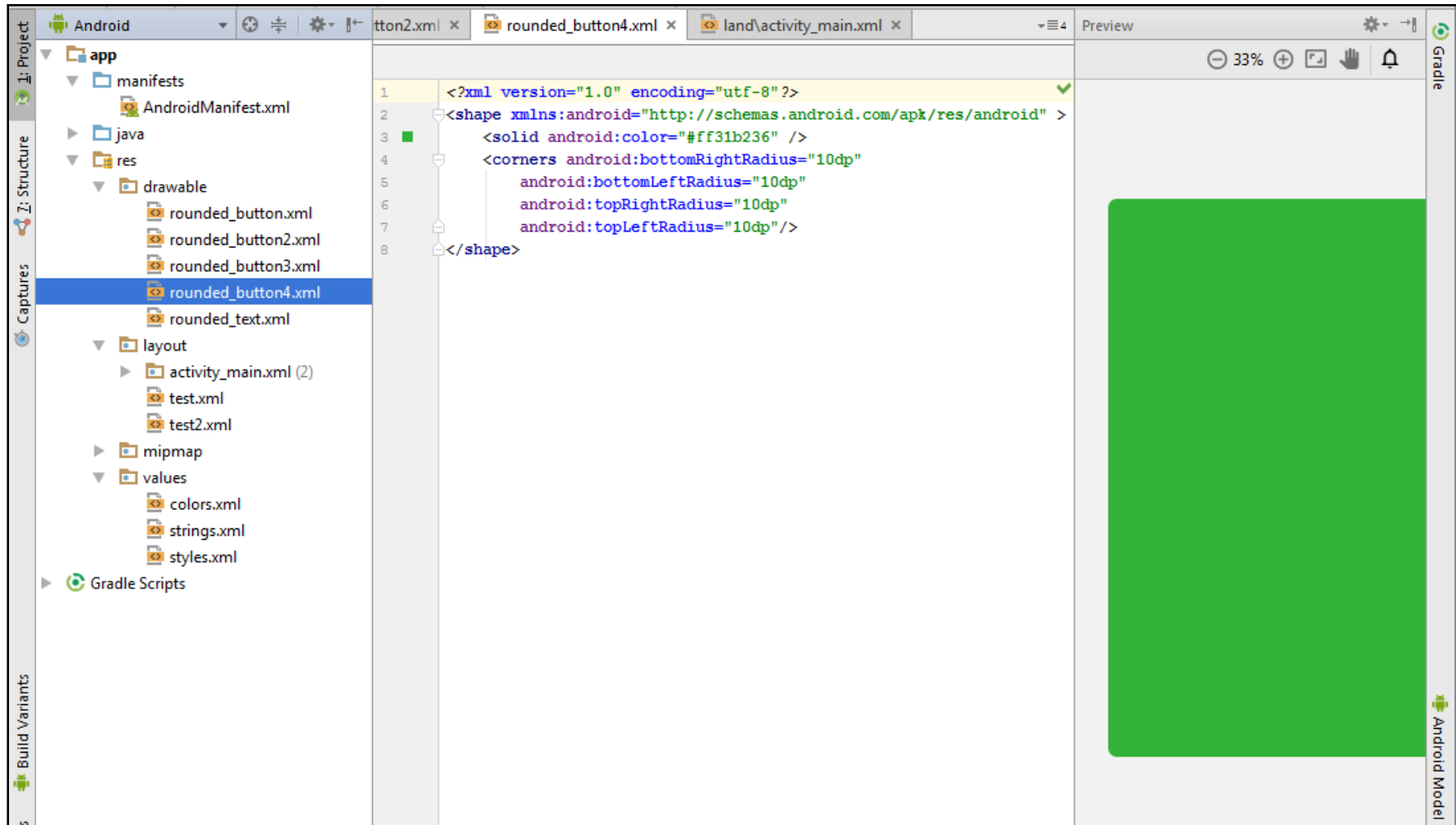
Подключаем стиль к компоненту в xml-файле графической разметки:



```
layout/activity_main.xml x land/activity_main.xml x styles.xml x
android.support.constraint.ConstraintLayout Button
1 <?xml version="1.0" encoding="utf-8"?>
2 <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res-auto"
3   xmlns:app="http://schemas.android.com/apk/res-auto"
4   xmlns:tools="http://schemas.android.com/tools"
5   android:layout_width="match_parent"
6   android:layout_height="match_parent"
7   app:layout_constraintHorizontal_weight="1"
8   tools:context="by.bsuir.lesson1.MainActivity">
9
10   <Button
11     android:id="@+id/buttonMC"
12     style="@style/ButtonStyleForHorizontalOrientation"
13     android:layout_marginEnd="8dp"
14     android:layout_marginLeft="8dp"
15     android:layout_marginRight="8dp"
16     android:layout_marginStart="8dp"
17     android:background="@drawable/rounded_button"
18     android:text="MC"
19     app:layout_constraintHorizontal_weight="1"
20     app:layout_constraintLeft_toLeftOf="parent"
21     app:layout_constraintRight_toLeftOf="@+id/buttonMR"
22     app:layout_constraintBaseline_toBaselineOf="@+id/buttonMR" />
23
24   <Button
25     android:id="@+id/buttonMR"
26     style="@style/ButtonStyleForHorizontalOrientation"
27     android:layout_marginEnd="8dp"
28     android:layout_marginLeft="8dp"
29     android:layout_marginRight="8dp"
30     android:layout_marginStart="8dp"
31     android:background="@drawable/rounded_button"
32     android:text="MR"
33     app:layout_constraintBaseline_toBaselineOf="@+id/buttonMS"
34     app:layout_constraintHorizontal_weight="1"
```

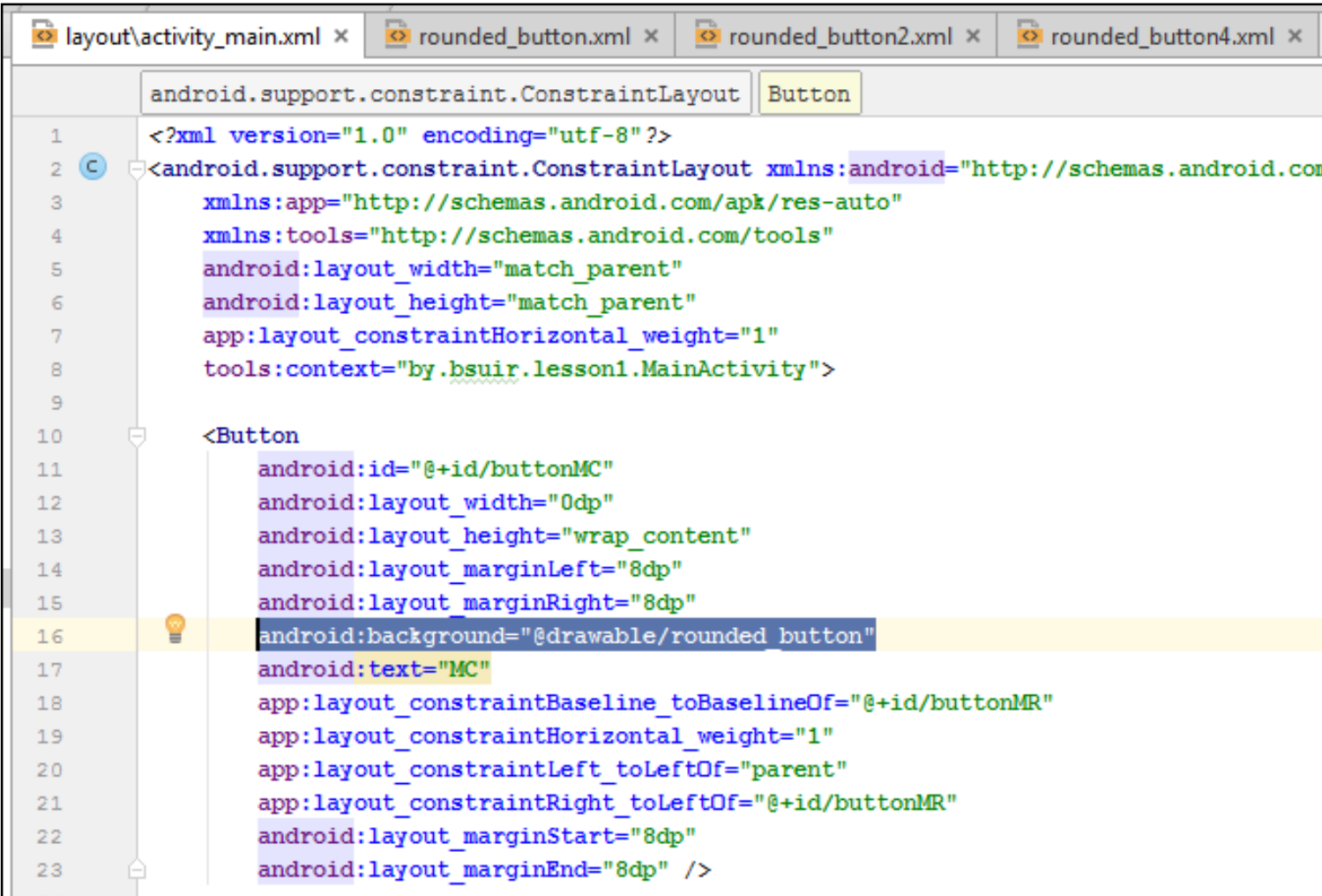
Рисуем свои кнопки:

Создаем шаблон в папке drawable:



Рисуем свои кнопки:

Подключаем шаблон к кнопке в xml-файле графической разметки:



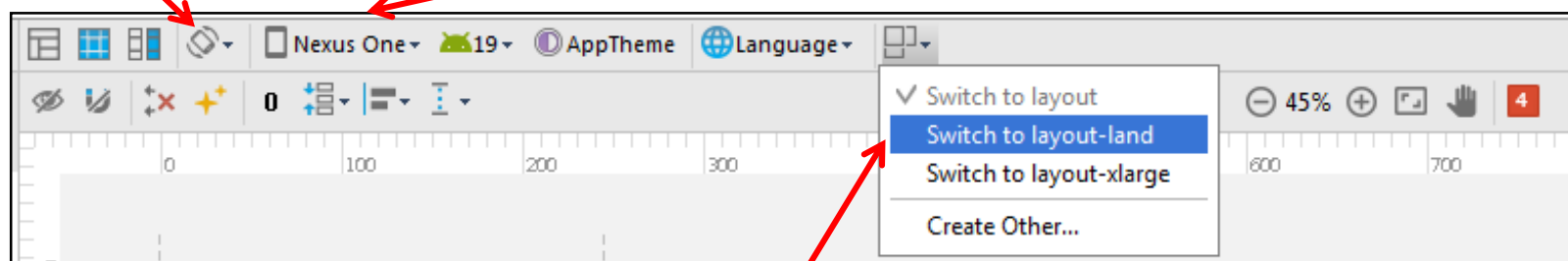
```
1 <?xml version="1.0" encoding="utf-8"?>
2 <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     app:layout_constraintHorizontal_weight="1"
8     tools:context="by.bsuir.lesson1.MainActivity">
9
10    <Button
11        android:id="@+id/buttonMC"
12        android:layout_width="0dp"
13        android:layout_height="wrap_content"
14        android:layout_marginLeft="8dp"
15        android:layout_marginRight="8dp"
16        android:background="@drawable/rounded_button"
17        android:text="MC"
18        app:layout_constraintBaseline_toBaselineOf="@+id/buttonMR"
19        app:layout_constraintHorizontal_weight="1"
20        app:layout_constraintLeft_toLeftOf="parent"
21        app:layout_constraintRight_toLeftOf="@+id/buttonMR"
22        android:layout_marginStart="8dp"
23        android:layout_marginEnd="8dp" />
```

Общие советы при работе с графикой

1. Сначала нарисуйте прототип графического интерфейса, чтобы четко представлять набор требуемых элементов и их взаимное расположение.
2. Определитесь с корневым layout (рекомендуемые: `ConstraintLayout`, `LinearLayout`).
3. Создавайте интерфейс сверху вниз.
4. Помните про адаптивный дизайн! Поэтому используйте весовые настройки, размеры в `dp`.
5. Создавайте свои стили в `styles.xml`, чтобы вынести отдельно и прописать общие свойства для целого набора ваших виджетов.
6. После размещения всех элементов дайте им уникальные и осмысленные `id`!!!

Общие советы при работе с графикой

7. Проверьте вид интерфейса на различных экранах, при повороте.



8. При необходимости сгенерируйте отдельный xml-файл с горизонтальным представлением (но только после завершения работы с вертикальным представлением, включая генерацию уникальных id) и переработайте взаимное расположение элементов.
9. Протестируйте на реальных устройствах или эмуляторах.