

Рис. 1.

При разработке, каждому Activity сопоставляется одноименный java-класс (наследник класса android.app.Activity). При запуске приложения, когда система должна показать Activity и в дальнейшем работать с ним, она будет вызывать методы этого класса. И от того, что мы в этих методах находим, зависит поведение Activity.

При создании модуля мы указывали, что надо создать Activity с именем MainActivity

Давайте посмотрим этот класс.

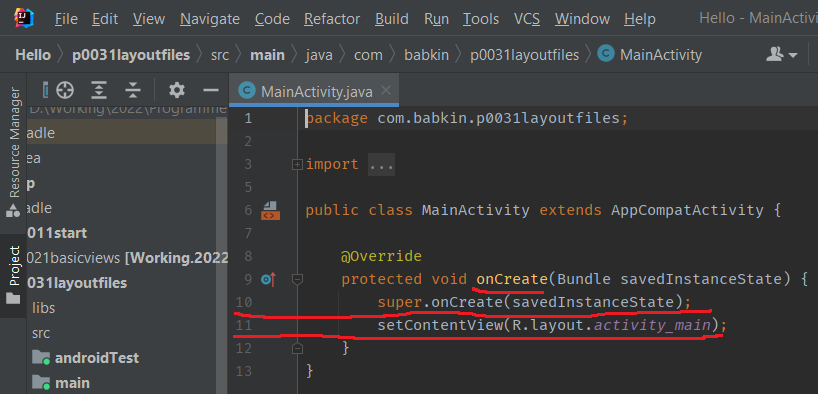


Рис. 2.

Смотрим java-код. Нас интересует метод onCreate – он вызывается, когда приложение создает и отображает Activity (на остальные методы пока не обращаем внимания). Посмотрим код реализации onCreate.

|  |  |
| --- | --- |
|  | 10 super.onCreate(savedInstanceState); |

это вызов метода родительского класса, выполняющий необходимые процедуры, его мы не трогаем.

Нас сейчас очень интересует следующая строка:

|  |  |
| --- | --- |
| 11 | setContentView(R.layout.activity\_main); |

Метод [setContentView(int)](http://developer.android.com/reference/android/app/Activity.html" \l "setContentView(int)" \t "_blank) – устанавливает содержимое Activity из layout-файла. Но в качестве аргумента мы указываем не путь к layout-файлу (res/layout/activity\_main.xml), а константу, которая является ID файла. Эта константа генерируется автоматически в файле R.java, который мы пока трогать не будем. В этом классе будут храниться сгенерированные ID для всех ресурсов проекта (из папки res/\*), чтобы мы могли к ним обращаться. Имена этих ID-констант совпадают с именами файлов ресурсов (без расширений).

Файл res/layout/activity\_main.xml был создан средой разработки вместе с Activity. Его название запрашивалось на том же экране, где и название Activity.

Попробуем отобразить содержимое другого файла. Создадим еще один layout-файл, например myscreen.xml. Для этого выделим папку res/layout в нашем модуле и нажмем на ней правую кнопку мыши. В появившемся меню выбираем New > Layout resource file. Для любителей горячих клавиш есть более удобный путь: при выделенной папке res/layout нажать ALT+Insert, и там уже Enter на пункте Layout resource file.

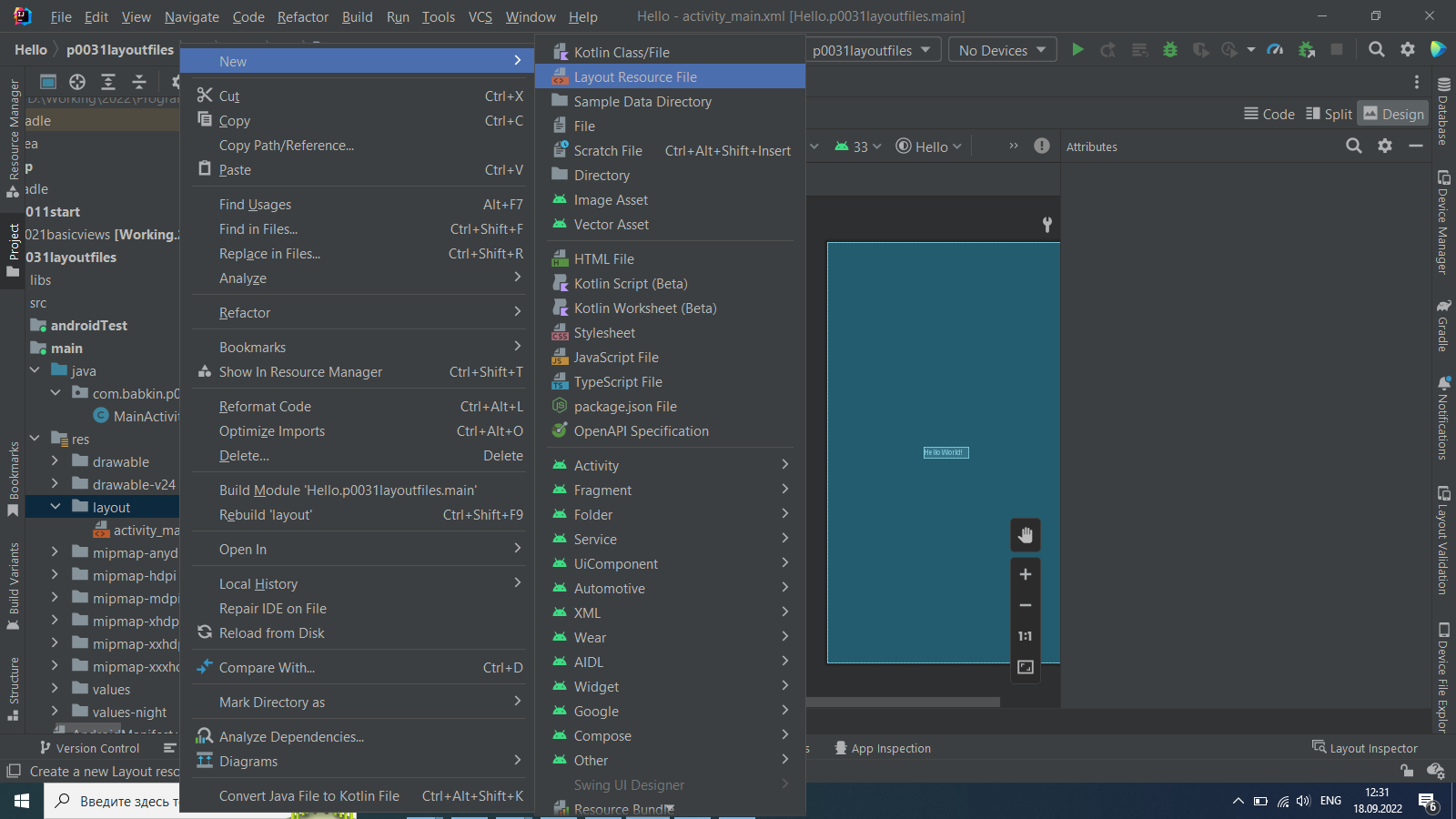


Рис. 3.

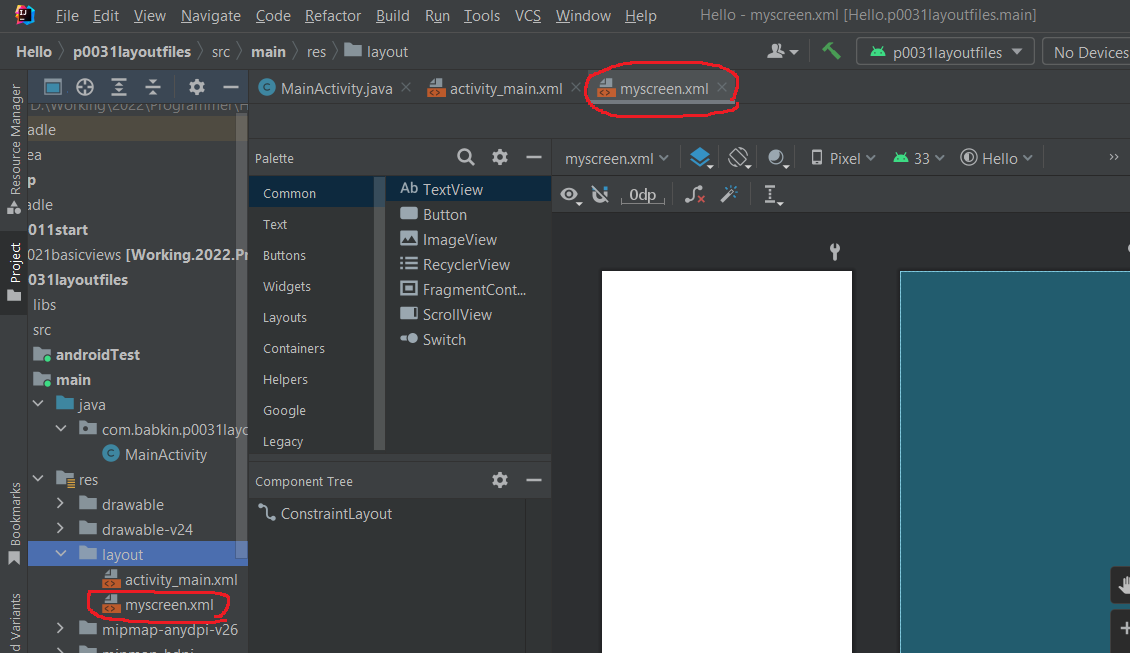


Рис. 4.

Добавим на экран элемент Plain TextView из списка слева и через Properties изменим его текст на: «new layout file myscreen for activity».

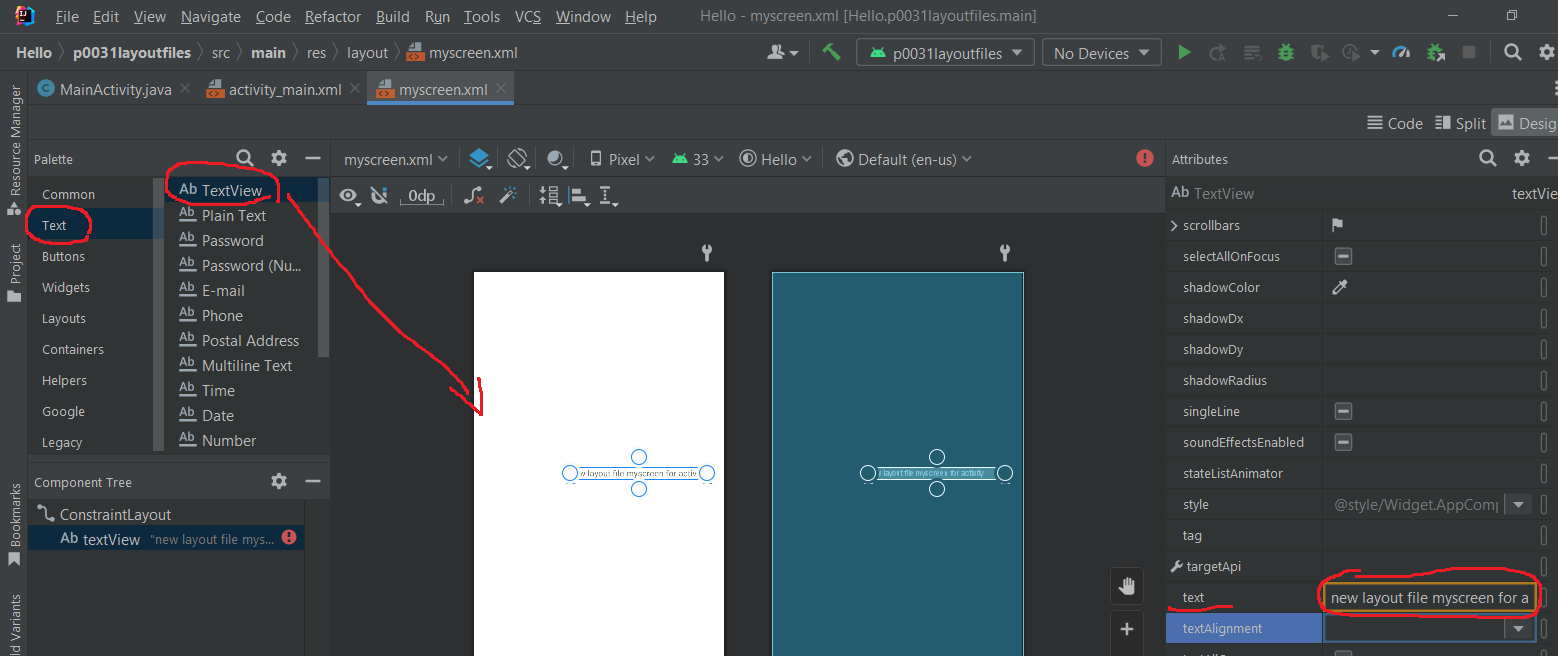


Рис. 5.

При создании нового layout-файла myscreen, среда добавила в R.java новую константу для этого файла - R.layout.myscreen. И мы теперь в коде сможем через эту константу указать на этот новый layout-файл.

Настроим так, чтобы Activity использовало новый файл myscreen.xml, а не activity\_main.xml, который был изначально. Откроем MainActivity.java и поменяем аргумент метода setContentView. Замените «R.layout.activity\_main», на «R.layout.myscreen» (ID нового layout-файла). Должно получиться так:

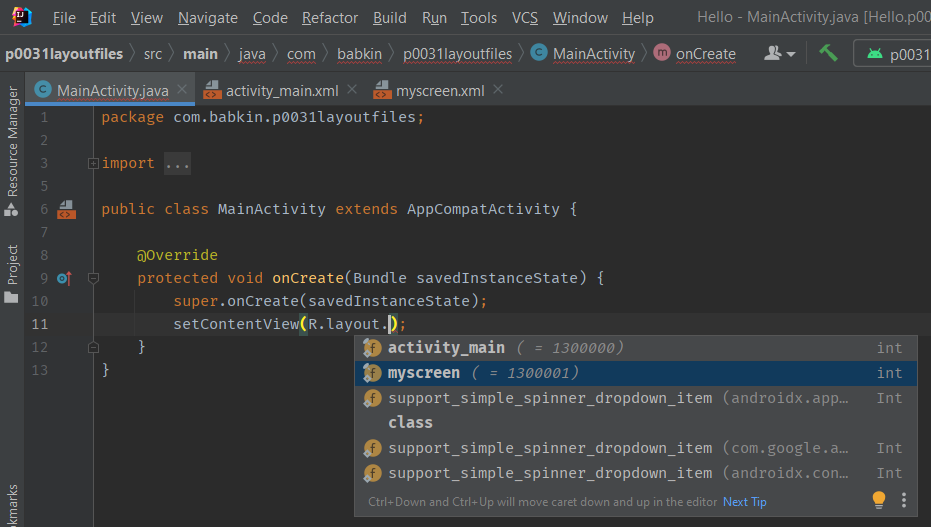


Рис. 6.



Рис. 7.

Видим, что теперь оно отображает содержимое из myscreen.xml, т.к. мы явно ему это указали в методе setContentView, который выполняется при создании (onCreate) Activity

Открыв в layout файл myscreen, вы видите его визуальное представление. Т.е. некий предпросмотр, как это будет выглядеть на экране. Откройте вкладку Code

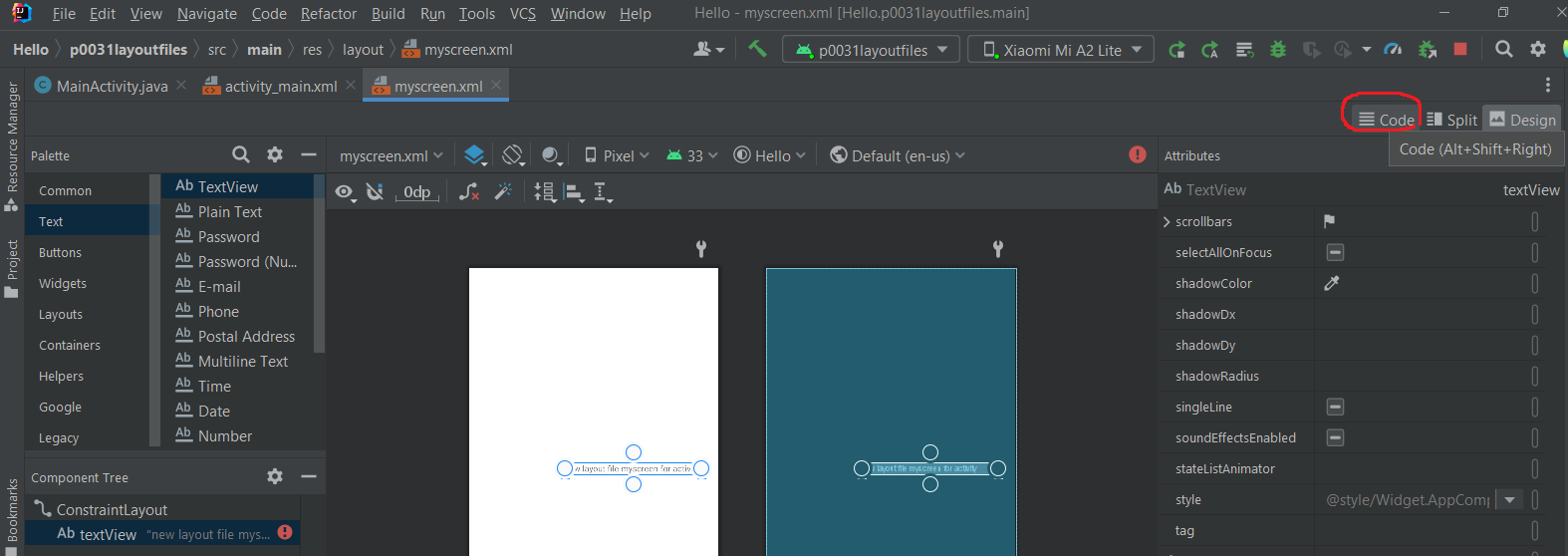


Рис. 8.

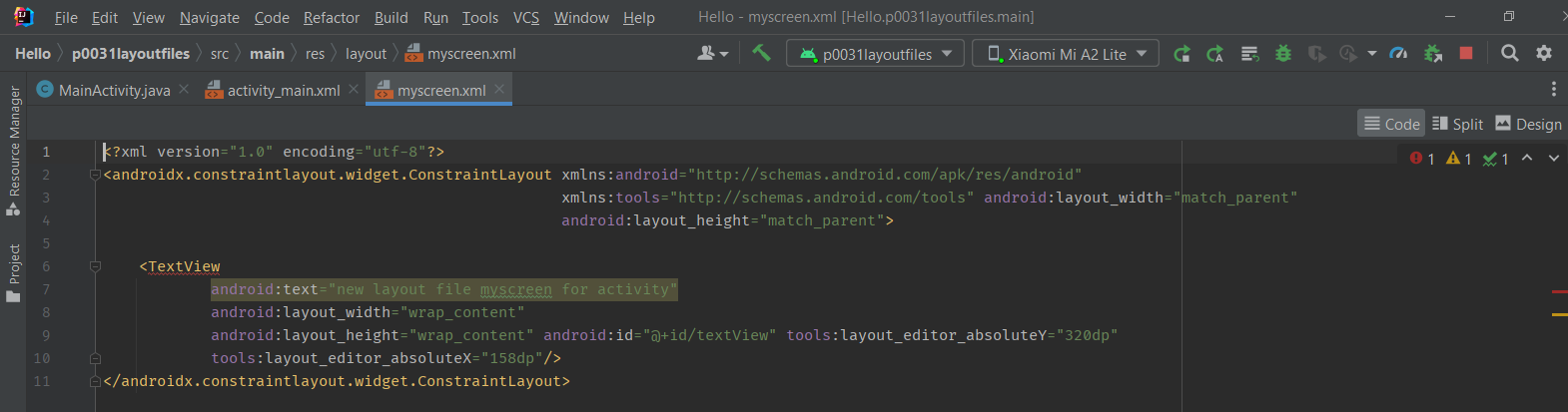


Рис. 9.

Мы видим достаточно читабельное xml-описание всех View нашего layout-файла. Названия xml-элементов - это классы View-элементов, xml-атрибуты - это параметры View-элементов, т.е. все те параметры, что мы меняем через вкладку Properties. Также вы можете вносить изменения прямо сюда и изменения будут отображаться во вкладке Design. Например, изменим текст у TextView. Вместо «new layout file myscreen for activity», напишем текст «Какой-либо текст»

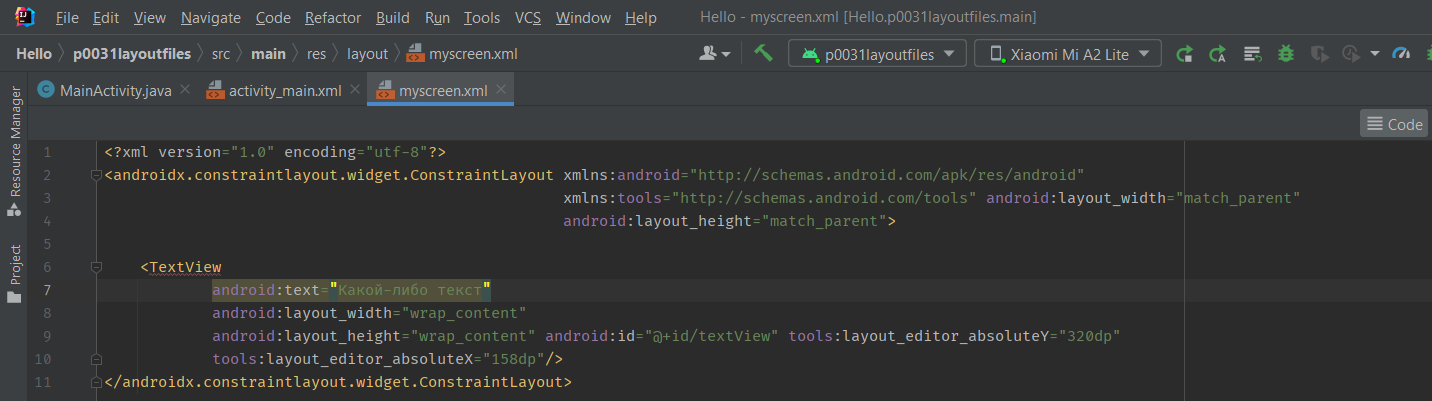


Рис. 10.

Сохраняем. Открываем Design и наблюдаем изменения.

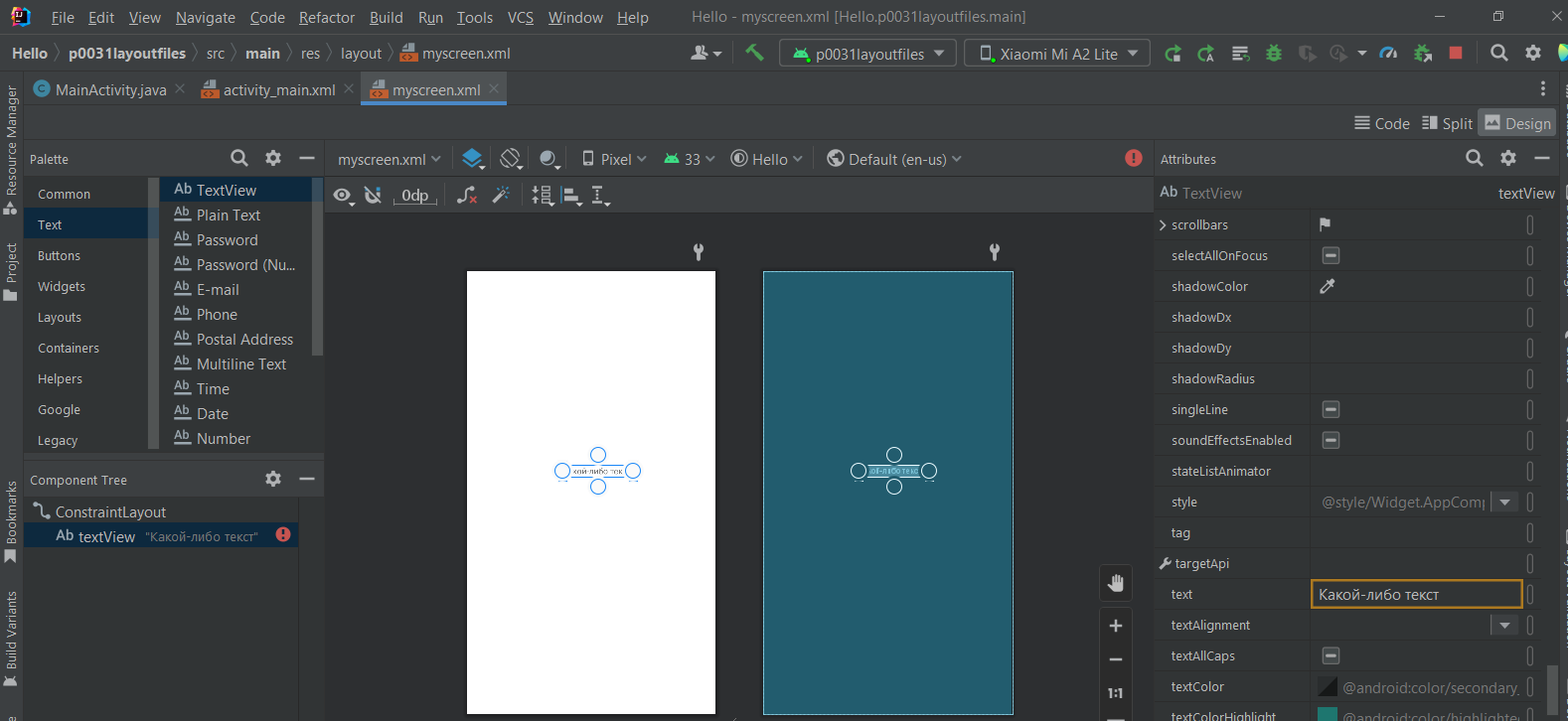


Рис. 11.

По умолчанию мы настраиваем layout-файл под вертикальную ориентацию экрана. Но что будет если мы повернем смартфон и включится горизонтальная ориентация? Давайте смотреть.

Изменим myscreen.xml. Добавим вертикальный ряд кнопок и изменим надпись.

xml-код (вы можете скопировать его и заменить им содержимое вашего layout файла myscreen во вкладке Text):

<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:id="@+id/textView1"

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="Вертикальная ориентация экрана">

    </TextView>

    <LinearLayout

        android:id="@+id/linearLayout1"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        android:orientation="vertical">

        <Button

            android:id="@+id/button1"

            android:layout\_width="100dp"

            android:layout\_height="100dp"

            android:text="Button1">

        </Button>

        <Button

            android:id="@+id/button2"

            android:layout\_width="100dp"

            android:layout\_height="100dp"

            android:text="Button2">

        </Button>

        <Button

            android:id="@+id/button3"

            android:layout\_width="100dp"

            android:layout\_height="100dp"

            android:text="Button3">

        </Button>

        <Button

            android:id="@+id/button4"

            android:layout\_width="100dp"

            android:layout\_height="100dp"

            android:text="Button4">

        </Button>

    </LinearLayout>

</LinearLayout>

Обратите внимание - я добавил вертикальный LinearLayout и поместил в него 4 кнопки. Подробнее обсудим это позднее.

Запустим приложение.

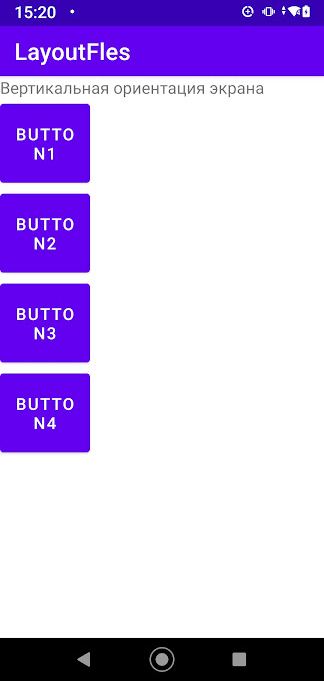


Рис. 12.

В вертикальной ориентации все ок. А в горизонтальной ориентации не все хорошо. Ориентация сменилась на горизонтальную и наши кнопки уже не влезают в экран.

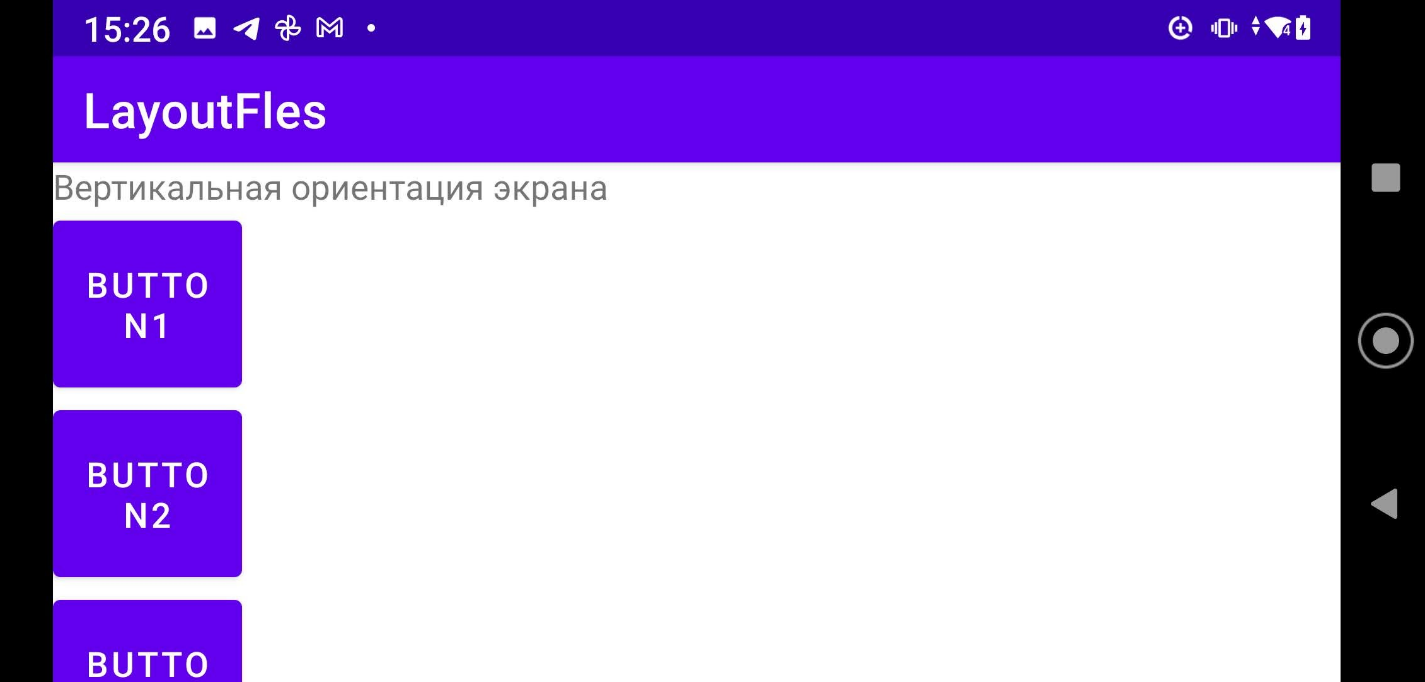


Рис. 13.

Т.е. нам необходим еще один layout-файл, который был бы заточен под горизонтальную ориентацию и в нашем случае вывел бы кнопки горизонтально.

Но как дать знать Activity, что она в вертикальной ориентации должна использовать один layout-файл, а в горизонтальной – другой? Об этом за нас уже подумали создатели Андроид. У нас есть возможность создать layout-файл, который будет использоваться приложением, когда устройство находится в горизонтальной ориентации. Создаем папку с именем layout-land, в которой создаем второй файл с именем myscreen

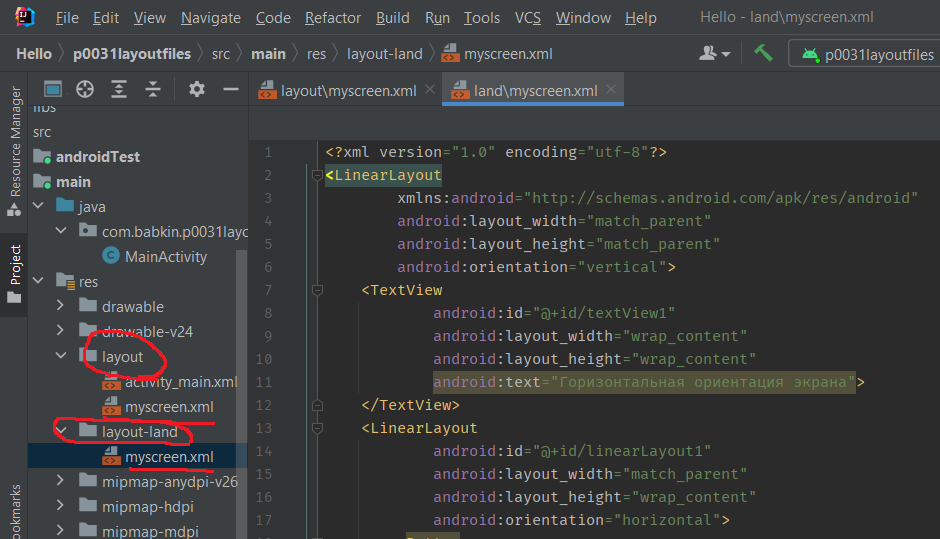


Рис. 14.

Во втором файле добавляем спецификатор, который даст приложению понять, что layout-файл надо юзать в горизонтальной ориентации. Нам надо указать, что нас интересует горизонтальная ориентация: Landscape. Выберите это значение из выпадающего списка.

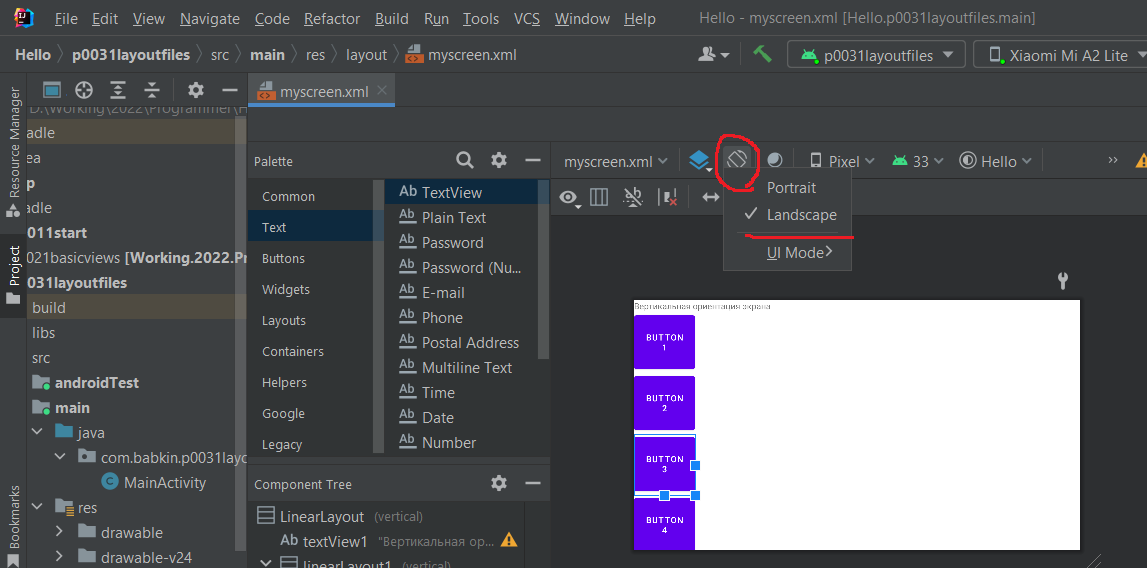


Рис. 15.