Создадим приложение «Уровень» на основе прошлой ЛБ про сенсор – акселерометр.

Сегодня будем использовать уже 2-а сенсора для получения боле точных значений – Accelerometer и Magnetic Field.

Начинаем работать в проекте прошлой лабораторной работы.

1. В MainActivity создаем несколько массивов.

Создаем массив magnetic в котором будет 9 элементов:

```
private var magnetic = FloatArray(9)
```

Создаем массив gravity в котором будет 9 элементов:

```
private var gravity = FloatArray(9)
```

Далее создаем 3-и массива с 3-мя элементами – данный об осях x, y и z:

```
    private var accrs = FloatArray(3)
    private var magf = FloatArray(3)
    private var values = FloatArray(3)
```

Массив accrs нужен чтобы записывать данные с акселерометра.

Массив magf нужен чтобы записывать данные с сенсора Magnetic Field. Для того чтобы более точно определять поворот экрана телефона будем использовать Magnetic Field.

Массив values необходим чтобы записывать готовые данные.

В итоге:

```
2
3   import ...
11
12  class MainActivity : AppCompatActivity() {
13      lateinit var sManager: SensorManager
14      private var magnetic = FloatArray( size: 9)
15      private var gravity = FloatArray( size: 9)
16
17      private var accrs = FloatArray( size: 3)
18      private var magf = FloatArray( size: 3)
19      private var values = FloatArray( size: 3)
20
21
22      override fun onCreate(savedInstanceState: Bundle?) {
23          super.onCreate(savedInstanceState)
24          setContentView(R.layout.activity_main)
25          val tvSensor = findViewById<TextView>(R.id.tvSensor)
26
27          sManager = getSystemService(Context.SENSOR_SERVICE) as SensorManager
```

Добавляем второй сенсор Magnetic Field:

```
val sensor2 = sManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD)
```

и регистрируем его:

```
sManager.registerListener(sListener, sensor2, SensorManager.SENSOR_DELAY_NORMAL)
```

Удаляем часть код из override fun onSensorChanged(sEvent: SensorEvent?) и поменяем имя с sEvent на event:

```
override fun onSensorChanged(event: SensorEvent?) {
```

И пишем в теле проверку когда получаем данные от акселерометра и когда получаем данные от Magnetic Field:

```
val sListener = object : SensorEventListener{
            override fun onSensorChanged(event: SensorEvent?) {
                when(event?.sensor?.type){
                    Sensor.TYPE_ACCELEROMETER-> accrs = event.values.clone()
                    Sensor.TYPE_MAGNETIC_FIELD-> magf = event.values.clone()
                }
```

Пишем RotationMatrix для вычисления поворота, куда передаем 4-е массива gravity, magnetic, accrs, magf:

```
                SensorManager.getRotationMatrix(gravity, magnetic, accrs, magf
```

Необходимо поменять координатную систему, т.е. сделать remapSystem.

Создаем временный массив outGravity, в котором будут данные те, которые нужны – в той координатной системе которая нужна:

```
                SensorManager.getRotationMatrix(gravity, magnetic, accrs, magf)
                val outGravity = FloatArray(9)
                SensorManager.remapCoordinateSystem(gravity,
                    SensorManager.AXIS_X,
                    SensorManager.AXIS_Z,
                    outGravity
                )
```

Эта функция сделает преобразование координатной системы и запишет в outGravity.

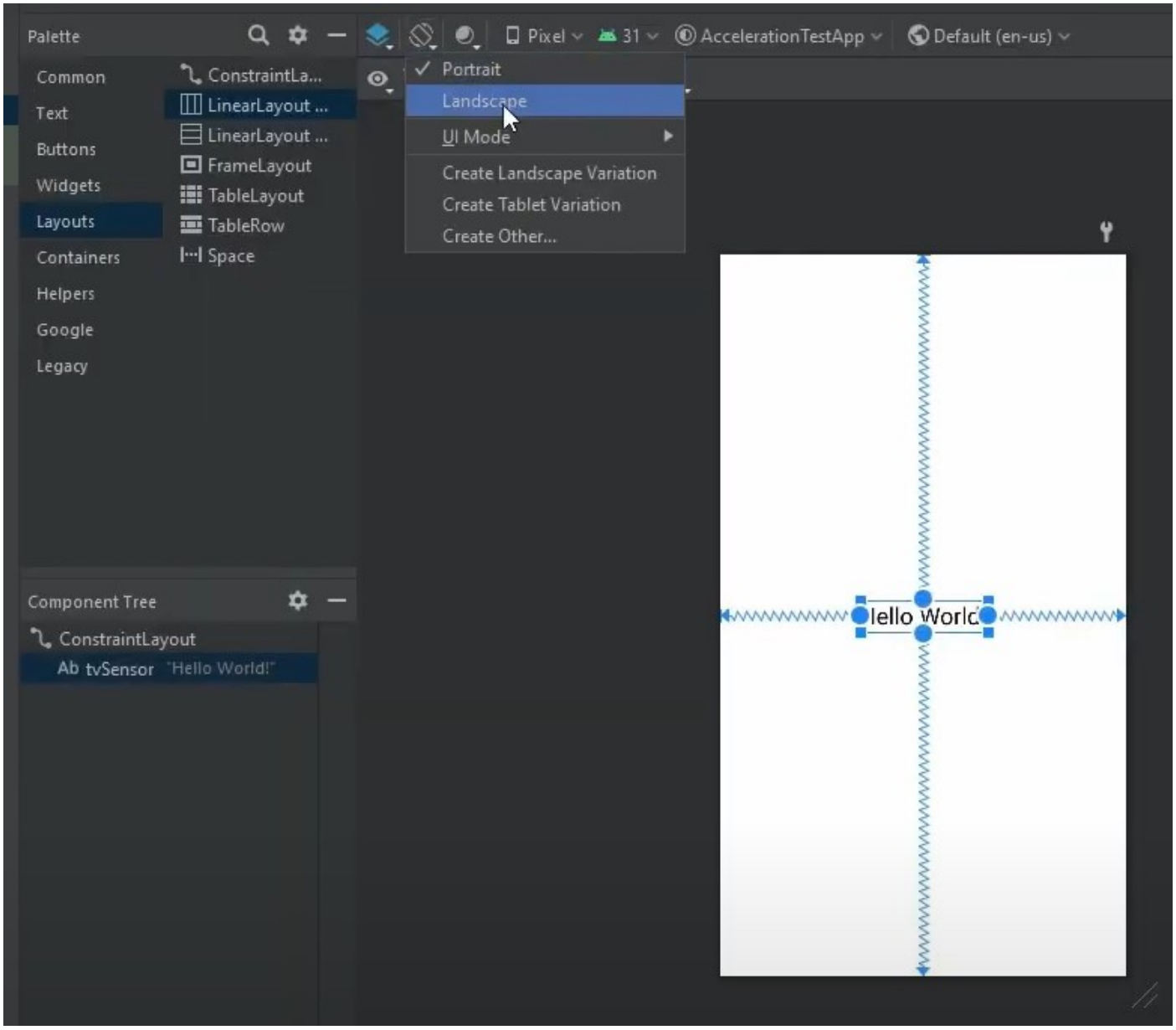
Теперь можем получить градусы наклона (в радианах пока):

```
SensorManager.getOrientation(outGravity, values)
```

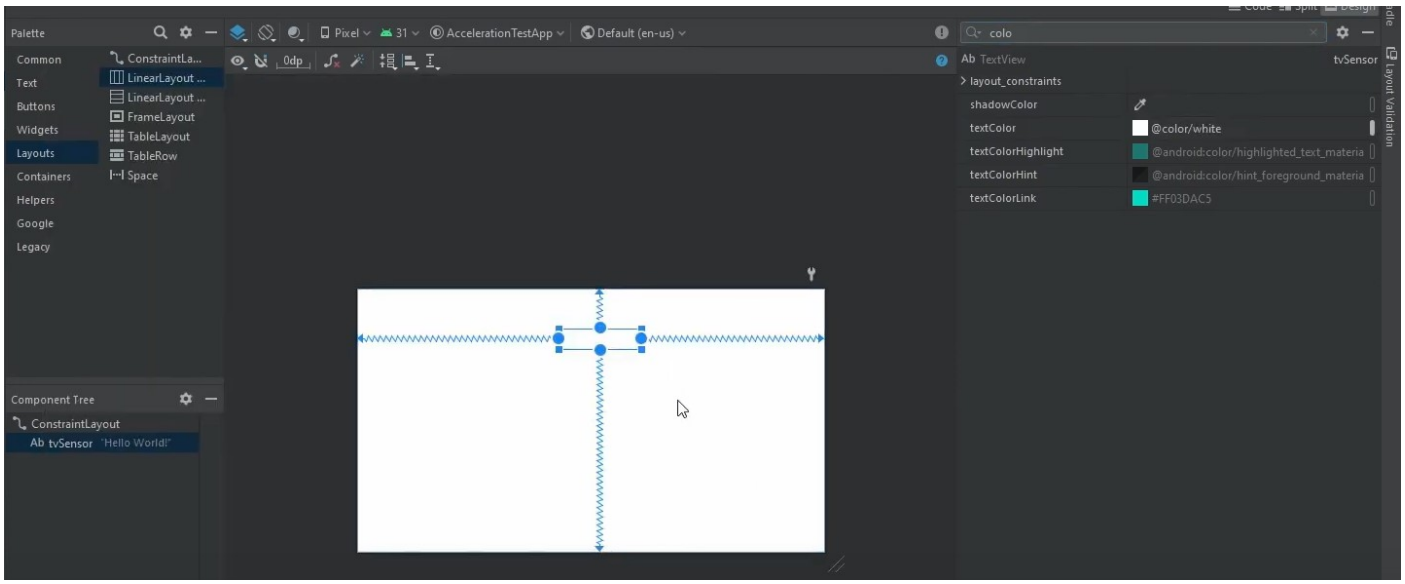
Передаем в функцию данные outGravity и получаем массив values, где на нулевой позиции будет угол поворота в радианах по оси X, на первой позиции будет угол поворота в радианах по оси Y, на второй позиции будет угол поворота в радианах по оси Z.

2. Переходим в activity_main.xml

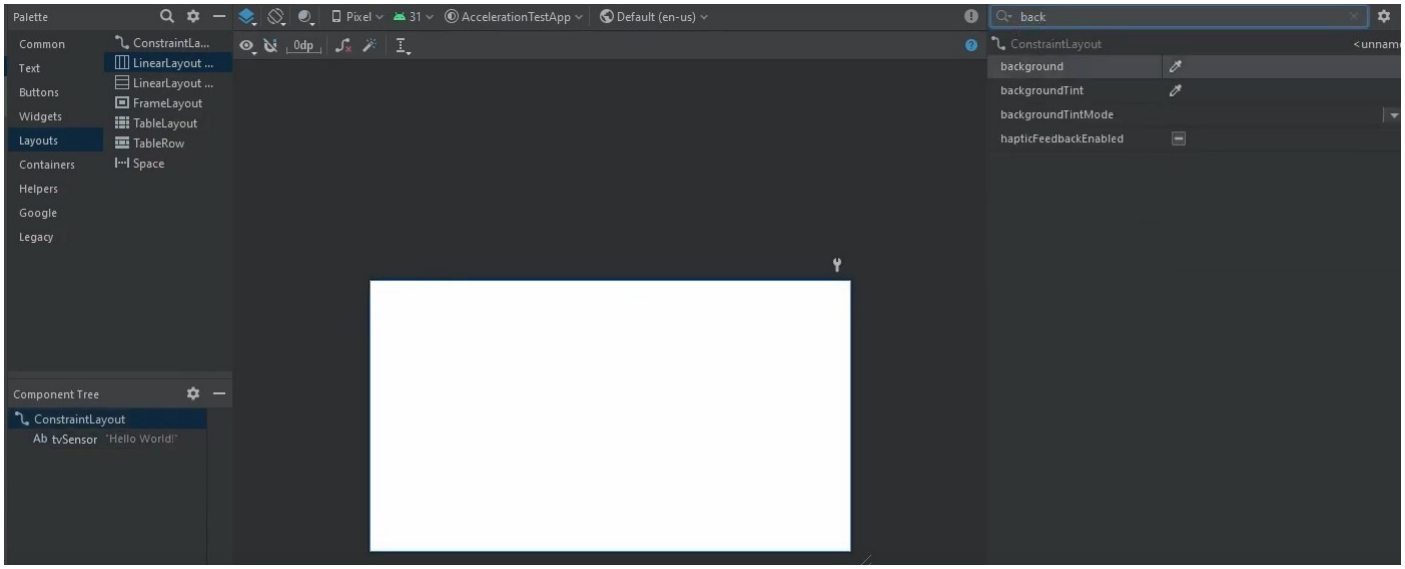
Выбирабем Landscape



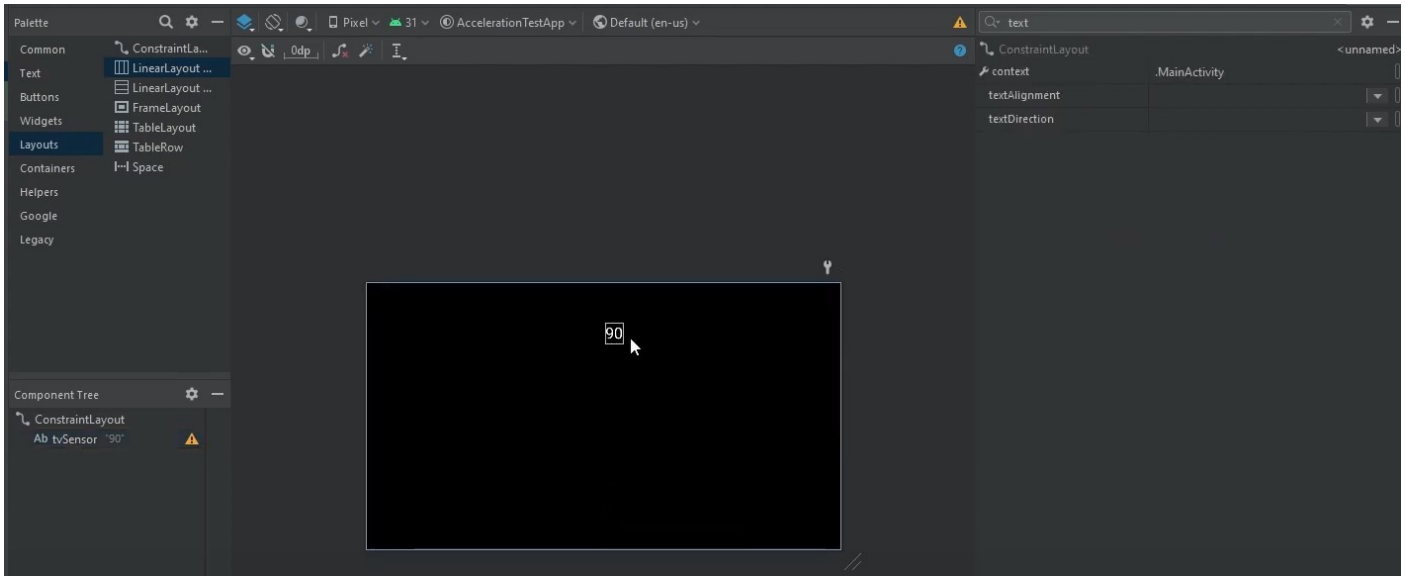
Меняем цвет tvSensor на белый:



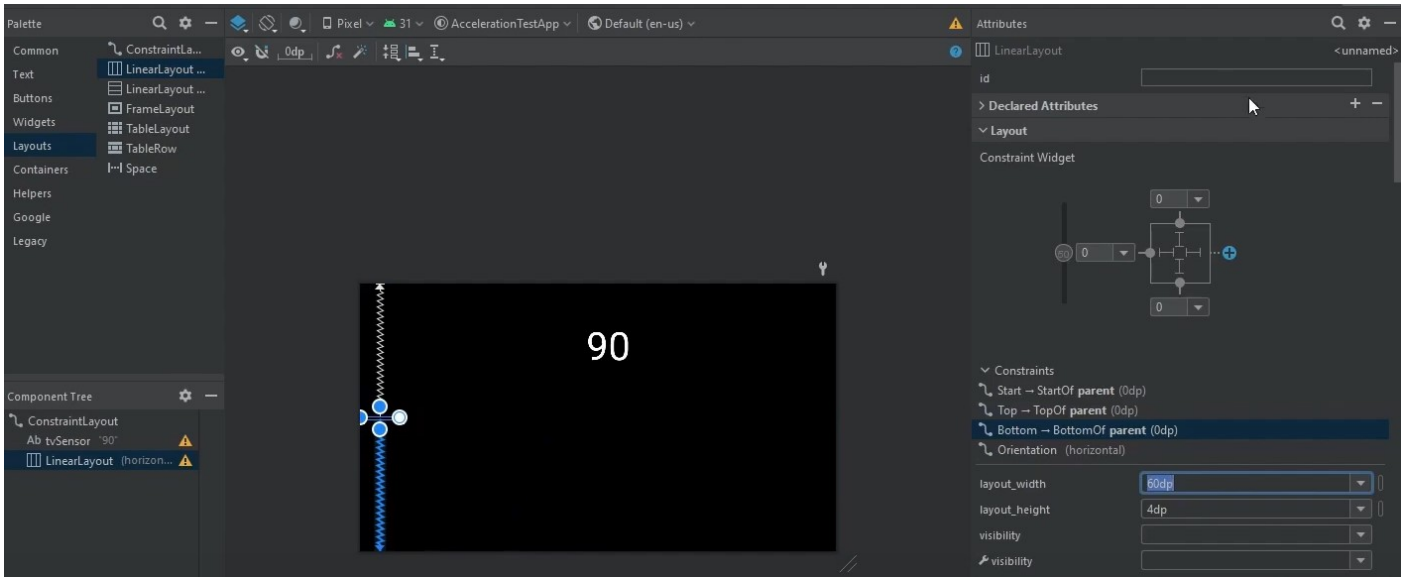
Фон ставим background черным:



Текст поменяем на «90» и размер 60sp:



Создаем LinearLayout, привязываем к верхнему и нижнему краям и выставляем ширину, высоту:

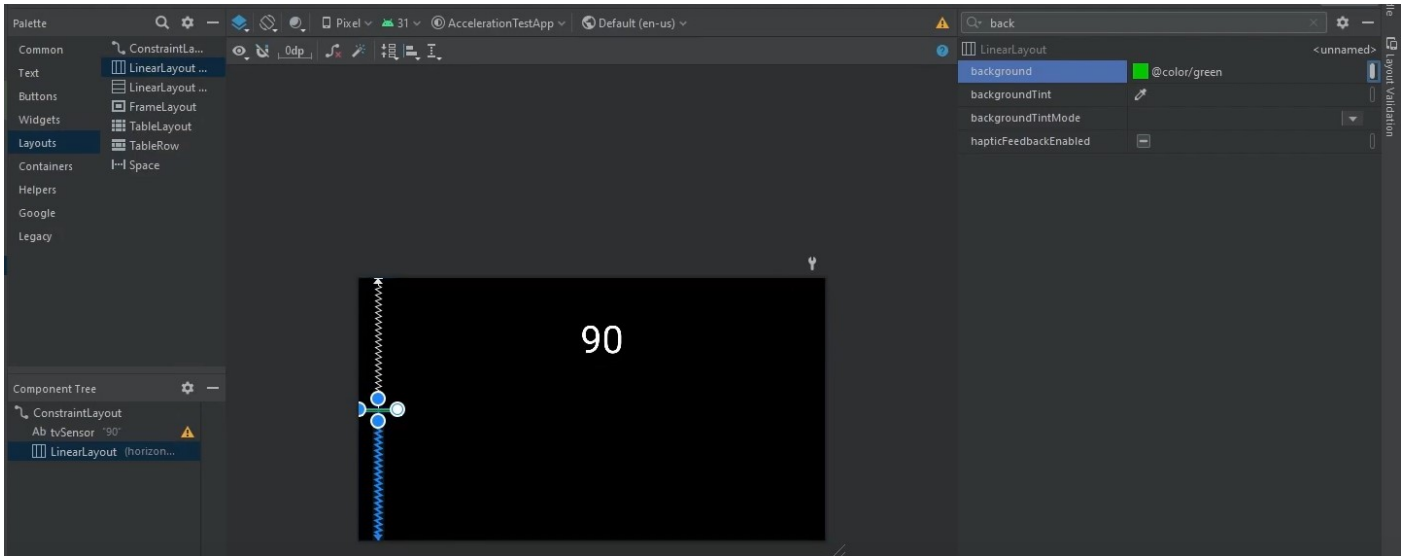


Создадим в colors.xml новый цвет – green:

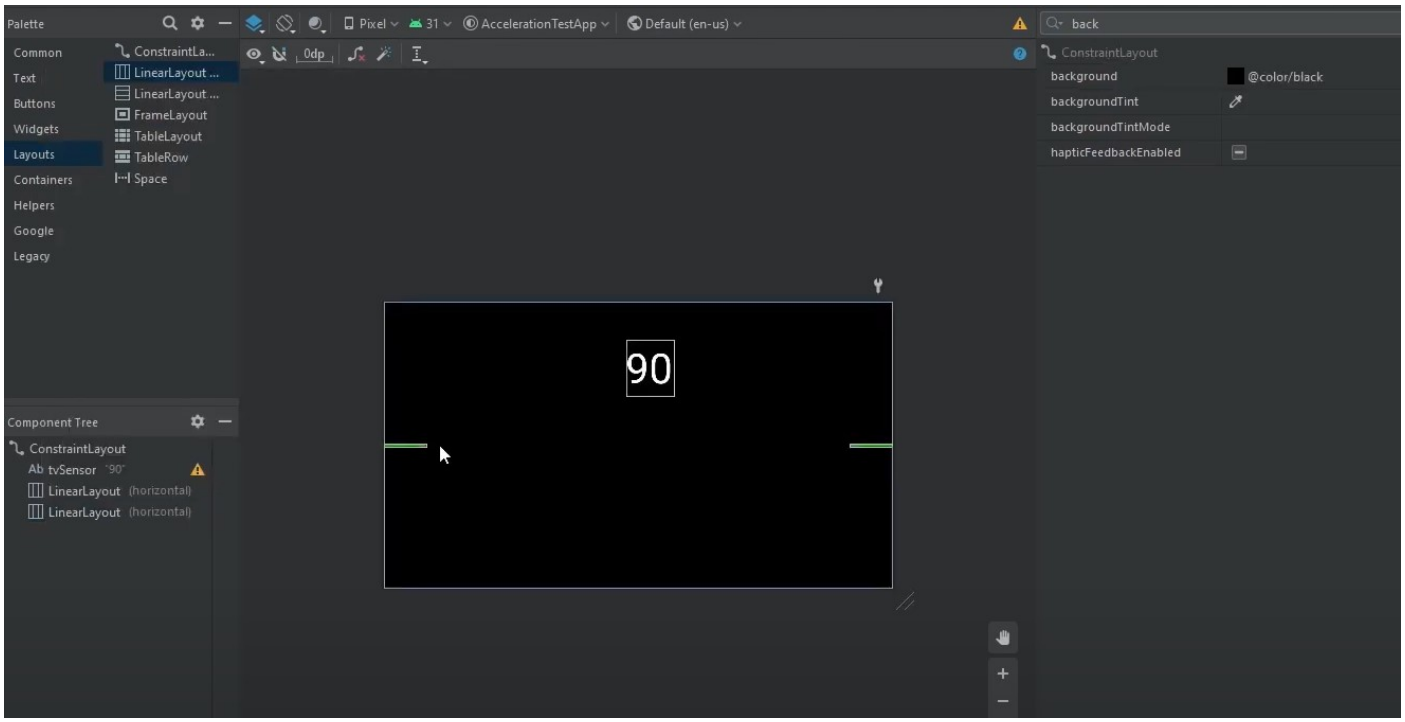
```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
    <color name="green">#07C500</color>
</resources>
```

И возвращаемся в activity_main.xml

Ставим цвет у LinearLayout background – green:

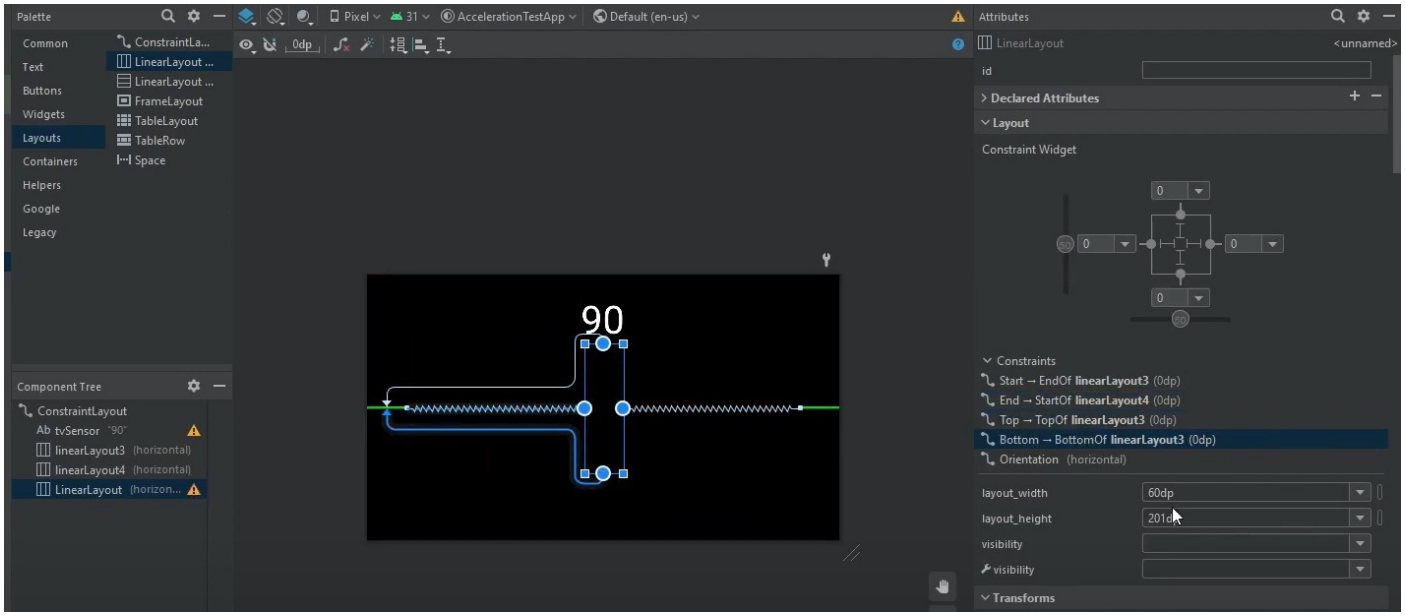


Делаем аналогичную «палочку» для правого края с такими же размерами и цветом.

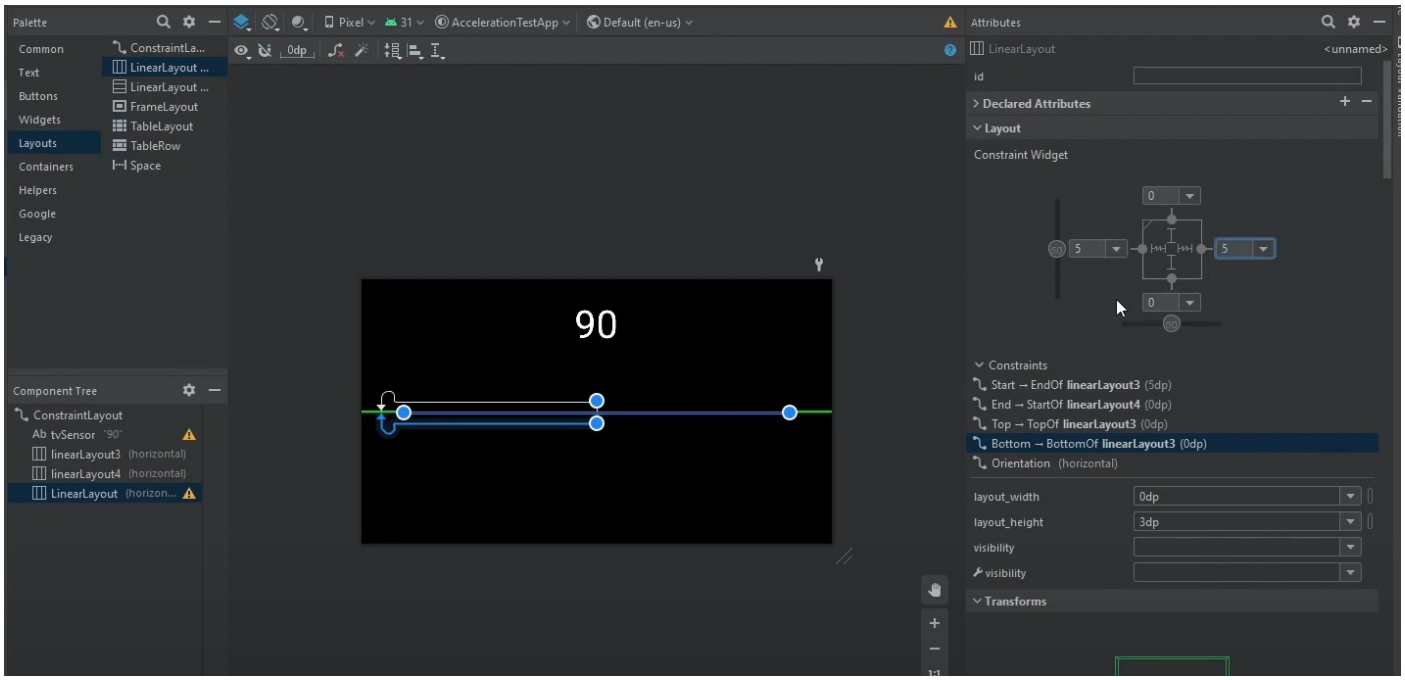


Создаем «палочку» посередине, которая будет крутиться, показывая уровень:

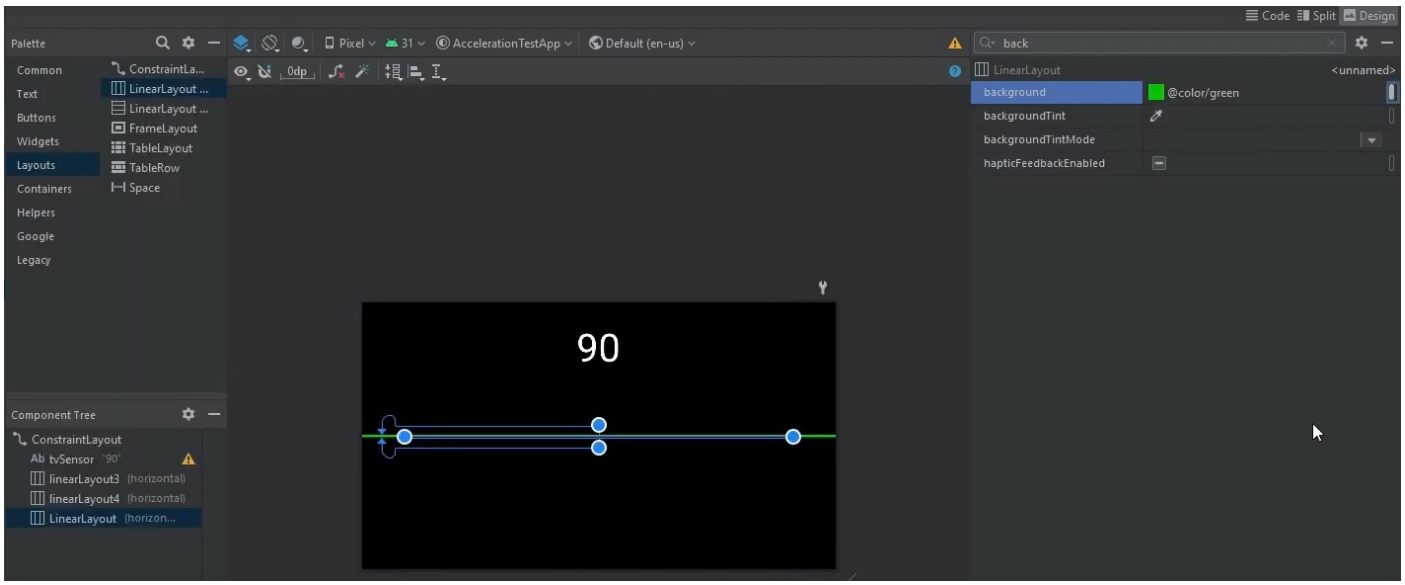
Перетаскиваем LinearLayout, привязываем в зеленым элементам (4-е привязки), выставляем размеры и цвет:



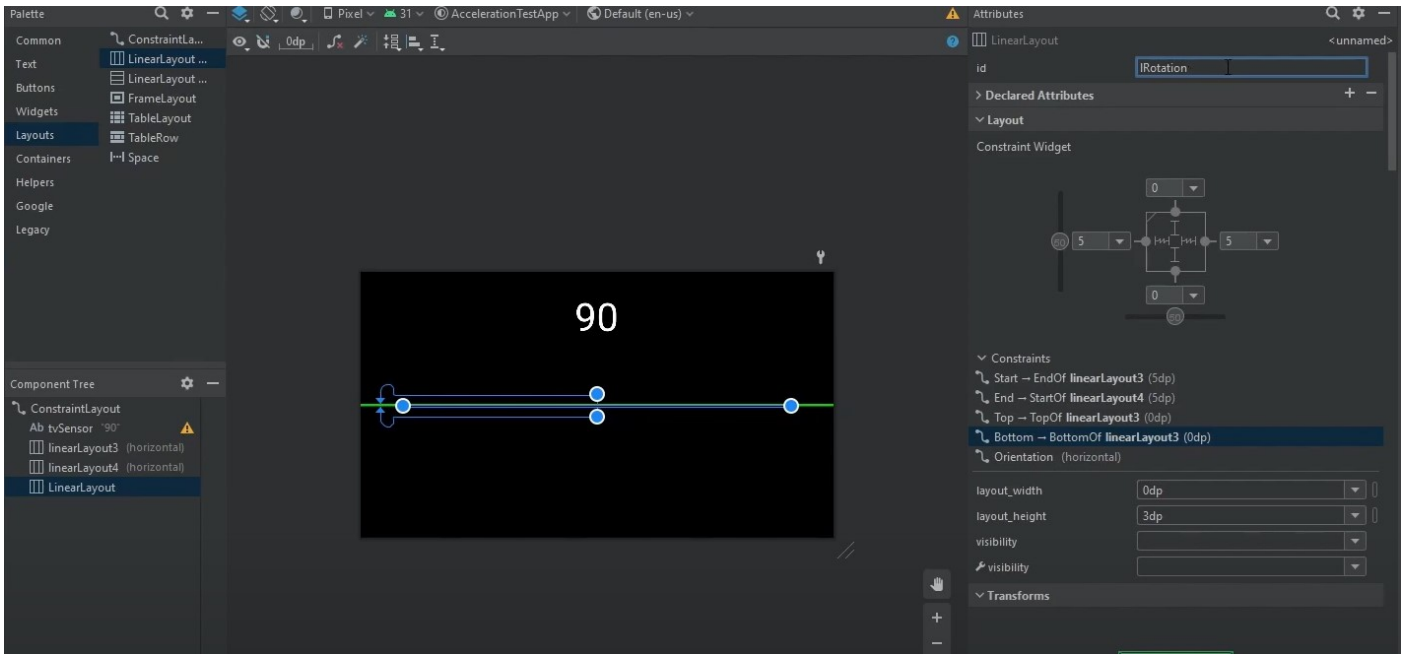
Меняем высоту, ширину и ставим отступы по 5 справа и слева:



Пока ставим цвет центрального элемента «зеленым». Потом в коде поставим возможность изменения этого центрального элемента – зеленый, если уровень, красный если нет уровня:



Поставим id этого центрального элемента как lRotatio:



3. Переходим в MainActivity.

Создаем переменную lRotation для того, чтобы найти lRotation:

```
val lRotation = findViewById<LinearLayout>(R.id.lRotation)
```

Переводим получаемые данные при повороте с радиан в градусы:

```
val degree = values[2] * 57.2958f
```

Указываем насколько необходимо повернуть:

```
val rotate = 270 + degree
```

```
lRotation.rotation = rotate
```

Для того чтобы ТексВью показывал «0» добавляем 90

```
val rData = 90 + degree

tvSensor.text = rData.toInt().toString()
```

Сделаем изменение цвета при «уровне»:

```
val color = if(rData.toInt() == 0){
    Color.GREEN
} else {
     Color.RED
}
```

Этот цвет в зависимости от поворота предаем в BackgroundColor:

```
lRotation.setBackgroundColor(color)
```

В итоге получаем код:

```
import android.content.Context
import android.graphics.Color
import android.hardware.Sensor
import android.hardware.SensorEvent
import android.hardware.SensorEventListener
import android.hardware.SensorManager
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.LinearLayout
import android.widget.TextView

class MainActivity : AppCompatActivity() {
    lateinit var sManager: SensorManager
    private var magnetic = FloatArray(9)
    private var gravity = FloatArray(9)

    private var accrs = FloatArray(3)
    private var magf = FloatArray(3)
    private var values = FloatArray(3)


    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        val tvSensor = findViewById<TextView>(R.id.tvSensor)
        val lRotation = findViewById<LinearLayout>(R.id.lRotation)

        sManager = getSystemService(Context.SENSOR_SERVICE) as SensorManager
        val sensor = sManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER)
        val sensor2 = sManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD)
        val sListener = object : SensorEventListener{
            override fun onSensorChanged(event: SensorEvent?) {
                when(event?.sensor?.type){
                    Sensor.TYPE_ACCELEROMETER-> accrs = event.values.clone()
                    Sensor.TYPE_MAGNETIC_FIELD-> magf = event.values.clone()
                }
```

```
                SensorManager.getRotationMatrix(gravity, magnetic, accrs, magf)
                val outGravity = FloatArray(9)
                SensorManager.remapCoordinateSystem(gravity,
                    SensorManager.AXIS_X,
                    SensorManager.AXIS_Z,
                    outGravity
                )
                SensorManager.getOrientation(outGravity, values)
                val degree = values[2] * 57.2958f
                val rotate = 270 + degree
                lRotation.rotation = rotate
                val rData = 90 + degree
                val color = if(rData.toInt() == 0){
                    Color.GREEN
                } else {
                    Color.RED
                }
                lRotation.setBackgroundColor(color)
                tvSensor.text = rData.toInt().toString()

            }

            override fun onAccuracyChanged(p0: Sensor?, p1: Int) {

            }

        }
        sManager.registerListener(sListener, sensor, SensorManager.SENSOR_DELAY_NORMAL)
        sManager.registerListener(sListener, sensor2, SensorManager.SENSOR_DELAY_NORMAL)
    }
}
```