



Activity\_main.xml

```
1 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2   xmlns:app="http://schemas.android.com/apk/res-auto"
3   xmlns:tools="http://schemas.android.com/tools"
4   android:id="@+id/main"
5   android:layout_width="match_parent"
6   android:layout_height="match_parent"
7   android:orientation="vertical"
8   android:background="@color/yellow"
9   tools:context=".MainActivity">
10
11   <Chronometer
12       android:id="@+id/textTime"
13       android:layout_width="wrap_content"
14       android:layout_height="wrap_content"
15       android:textSize="80sp"
16       android:layout_gravity="center"
17   />
18   <Button
19       android:id="@+id/btnStart"
20       android:layout_width="wrap_content"
21       android:layout_height="wrap_content"
22       android:text="@string/start"
23       android:layout_gravity="center"
24       android:textColor="@color/black"
25   />
26   <Button
27       android:id="@+id/btnPause"
28       android:layout_width="wrap_content"
29       android:layout_height="wrap_content"
30       android:text="@string/pause"
31       android:layout_gravity="center"
32       android:textColor="@color/black"
33   />
34   <Button
35       android:id="@+id/btnReset"
36       android:layout_width="wrap_content"
```

```
    <Button
        android:id="@+id/btnReset"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/reset"
        android:layout_gravity="center"
        android:textColor="@color/black"
    />
</LinearLayout>
```

String.xml

```
1  <resources>
2      <string name="start">start</string>
3      <string name="pause">pause</string>
4          <string name="reset">reset</string>
5      <string name="app_name">string\n\n</string>
6  </resources>
```

MainActivity.kt

```
1 package com.example.chronometr
2
3 import android.os.Bundle
4 import android.os.SystemClock
5 import android.widget.Button
6 import android.widget.Chronometer
7 import androidx.activity.enableEdgeToEdge
8 import androidx.appcompat.app.AppCompatActivity
9 import androidx.core.view.ViewCompat
10 import androidx.core.view.WindowInsetsCompat
11
12 class MainActivity : AppCompatActivity() {
13     lateinit var chronometer: Chronometer
14     var running = false
15     var offset: Long = 0
16
17     // Константы для ключей в Bundle
18     private val OFFSET_KEY = "offset"
19     private val RUNNING_KEY = "running"
20     private val BASE_KEY = "base_key"
21
22     override fun onCreate(savedInstanceState: Bundle?) {
23         super.onCreate(savedInstanceState)
24         enableEdgeToEdge()
25         setContentView(R.layout.activity_main)
26
27         // Инициализация элементов View
28         chronometer = findViewById(R.id.textTime)
29         val btnStart = findViewById<Button>(R.id.btnStart)
30         val btnPause = findViewById<Button>(R.id.btnPause)
31         val btnReset = findViewById<Button>(R.id.btnReset)
32
33         // Обработка нажатий
34         btnStart.setOnClickListener {
35             if (!running) {
36                 setBaseTime()
37                 chronometer.start()
```

```

38         running = true
39     }
40 }
41
42 btnPause.setOnClickListener {
43     if (running) {
44         saveOffset()
45         chronometer.stop()
46         running = false
47     }
48 }
49
50 btnReset.setOnClickListener {
51     offset = 0
52     setBaseTime()
53     running = false
54 }
55
56 // Восстановление состояния при создании активности (например, после поворота)
57 if (savedInstanceState != null) {
58     offset = savedInstanceState.getLong(OFFSET_KEY)
59     running = savedInstanceState.getBoolean(RUNNING_KEY)
60     chronometer.base = savedInstanceState.getLong(BASE_KEY)
61     if (running) {
62         chronometer.start()
63     } else {
64         setBaseTime()
65     }
66 }
67 }
68
69 // Сохранение состояния перед уничтожением активности
70 @Override fun onSaveInstanceState(outState: Bundle) {

```

```

// Сохранение состояния перед уничтожением активности
@Override fun onSaveInstanceState(outState: Bundle) {
    super.onSaveInstanceState(outState)
    outState.putBoolean(RUNNING_KEY, running)
    outState.putLong(OFFSET_KEY, offset)
    outState.putLong(BASE_KEY, chronometer.base)
}

private fun saveOffset() {
    offset = SystemClock.elapsedRealtime() - chronometer.base
}

private fun setBaseTime() {
    chronometer.base = SystemClock.elapsedRealtime() - offset
}
}

```