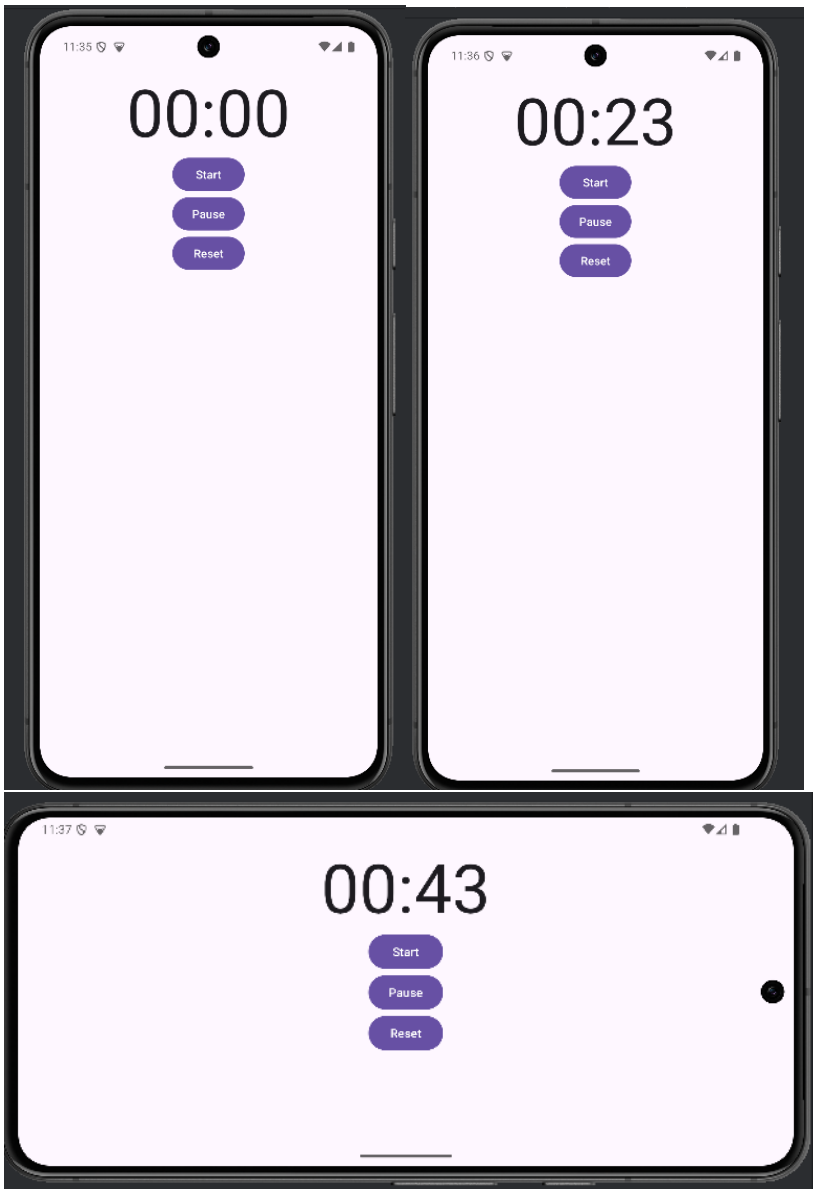
Готовое приложение:



Код xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <Chronometer
        android:id="@+id/textTime"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="80sp"
        android:layout_gravity="center"
        />
    <Button
        android:id="@+id/btnStart"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:text="@string/start"
        android:layout_gravity="center"
        />
    <Button
        android:id="@+id/btnPause"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/pause"

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

Код kt:

```
package com.example.sekundomer_ruslan

import android.os.Bundle
import android.os.SystemClock
import android.widget.Button
import android.widget.Chronometer
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class MainActivity : AppCompatActivity() {

    // Переменные для хранения состояния хронометра
    lateinit var chronometr: Chronometer // Хронометр
    var running: Boolean = false // Хронометр работает?
    var offset: Long = 0 // Базовое смещение

    // Добавление строк для ключей, используемых с Bundle
    val OFFSET_KEY = "offset"
    val RUNNING_KEY = "running"
    val BASE_KEY = "base_key"

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)

        // Настройка inset'ов для корректного отображения с edge-to-edge
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom)
            insets
        }

        // Объявление переменных
        chronometr = findViewById(R.id.textTime)
        val btnStart = findViewById<Button>(R.id.btnStart)
```

```
        val btnPause = findViewById<Button>(R.id.btnPause)
        val btnReset = findViewById<Button>(R.id.btnReset)

        // Восстановление предыдущего состояния
        if (savedInstanceState != null) {
            offset = savedInstanceState.getLong(OFFSET_KEY)
            running = savedInstanceState.getBoolean(RUNNING_KEY)
            if (running) {
                chronometr.base = savedInstanceState.getLong(BASE_KEY)
                chronometr.start()
            } else {
                setBaseTime()
            }
        } else {
            setBaseTime() // Установка базового времени при первом запуске
        }

        // Обработчики нажатий для кнопок
        btnStart.setOnClickListener {
            if (!running) {
                setBaseTime() // Устанавливаем базовое время при запуске
                chronometr.start()
                running = true
            }
        }

        btnPause.setOnClickListener {
            if (running) {
                saveOffset()
                chronometr.stop()
                running = false
            }
        }

        btnReset.setOnClickListener {
            chronometr.stop()
            offset = 0  // Сбрасываем offset при обнулении
            setBaseTime() // Устанавливаем базовое время на текущий момент
            running = false
        }
    }

    override fun onSaveInstanceState(savedInstanceState: Bundle) {
        super.onSaveInstanceState(savedInstanceState)
        savedInstanceState.putLong(OFFSET_KEY, offset)
        savedInstanceState.putBoolean(RUNNING_KEY, running)
        savedInstanceState.putLong(BASE_KEY, chronometr.base)
    }

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

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