МИНИCTEPCTBO НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

**Кафедра прикладной информатики**

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6

ДИСЦИПЛИНЫ «Программирование мобильных устройств»

НА ТЕМУ:

**Таймеры и секундомеры. Логирование.**

**Выполнил:**

студент группы ПИН-б-о-21-1

Стороженко Артем Владимирович

Проверил: А. А. Щеголев

Ставрополь, 2023

**Цель работы:** изучить принцип работы логов для упрощения процесса разработки, а также научиться использовать виджеты, отвечающие за отсчет времени в android, а именно chronometer и TimerTask.

**Ход работы**

В рамках лабораторной работы №6 , вариант 8, необходимо было реализовать использование виджета chronometer (секундомер), для вычисления значения функции F(x) для каждого тика секундометра (x принимает значения от -30 до 30 в течение минуты).

В начале, был разработан интерфейс приложения, содержащий формулу, поля для ввода параметров функции, сам секундомер и три кнопки, отвечающие за старт, остановку и продолжение работы секундомера. Внешний вид приложения показан на рисунке 1.



Рисунок 1 – Внешний вид приложения для задания 6

Далее, были описаны обработчики событий кликов на кнопки. Кнопка старт начинает отсчет с начала, кнопка стоп – останавливает его, сохраняя значение, кнопка продолжить – начинает отсчет времени остановки.

Кнопки стоп и продолжить изначально заблокированы, кнопка стоп становиться активной после нажатия на кнопку старт, кнопка продолжить – после клика на кнопку стоп. Также, во время работы таймера изменение полей для параметров запрещается. Изменять их можно либо при остановленном, либо при неработающем таймере. Код обработчиков кнопок представлен на рисунке 2.

public void startChron(View v) {  
 chron.setBase(SystemClock.*elapsedRealtime*());  
 x = -30;  
 chron.start();  
 count\_flag = true;  
 buttonResume.setEnabled(false);  
 buttonStop.setEnabled(true);  
 a\_view.setEnabled(false);  
 b\_view.setEnabled(false);  
 c\_view.setEnabled(false);  
}  
  
public void stopChron(View v) {  
 count\_flag = false;  
 lastPause = SystemClock.*elapsedRealtime*();  
 chron.stop();  
 buttonResume.setEnabled(true);  
 a\_view.setEnabled(true);  
 b\_view.setEnabled(true);  
 c\_view.setEnabled(true);  
}  
  
public void resumeChron(View v) {  
 count\_flag = false;  
 chron.setBase(chron.getBase() + SystemClock.*elapsedRealtime*() - lastPause);  
 chron.start();  
 count\_flag = true;  
 buttonResume.setEnabled(false);  
 a\_view.setEnabled(false);  
 b\_view.setEnabled(false);  
 c\_view.setEnabled(false);  
}

Рисунок 2 – Обработчики кнопок

Далее, была описана функция F, которая получает значения из полей приложения, выбирает нужную формулу в соответствии с условиями и возвращает значение. Код функции F показан на рисунке 3.

public double F(double x)  
{  
 a = Double.*parseDouble*(a\_view.getText().toString());  
 b = Double.*parseDouble*(b\_view.getText().toString());  
 c = Double.*parseDouble*(c\_view.getText().toString());  
 if (c < 0 && b != 0)  
 {  
 return (a\*x\*x + b\*b\*x);  
 }  
 else if (c > 0 && b == 0)  
 {  
 return ((x+a)/(x+c));  
 }  
 else  
 {  
 return x / c;  
 }  
}

Рисунок 3 – Код функции F

Наконец, был создан слушатель для секундомера, отвечающий за изменение значения x его вывода на экран. Его код представлен на рисунке 4.

@Override  
 public void onChronometerTick(Chronometer chronometer) {  
 if (count\_flag) {  
 if (x == 30) {  
 x = -30;  
 } else {  
 x += 1;  
 }  
 }  
 value\_string = "F(" + x + ") = ";  
 x\_view.setText(value\_string);  
 value\_string = String.*format*(Locale.*ENGLISH*,"%,.4f", F(x));  
 value\_view.setText(value\_string);  
 }  
});

Рисунок 4 – Код слушателя для секундомера

**Вывод:** в результате выполнения задания лабораторной работы я научился использовать логирование для отладки приложения, а также работать с виджетами TimerTask и Chronometer, в частности, использовать слушатель их тиков.