Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по программе «**Практическое занятие 19**»

Выполнил: Рявкин Лев Алексеевич

Группа: ПР-22

Преподаватель: Мирошниченко Г.В

2023

**Задание:** 1 Используя UI-фрагменты и FragmentManager, начать строить приложение

CriminalIntent. Оно предназначено для хранения информации об «офисных

преступлениях»: грязной посуде, оставленной в раковине, или пустом лотке общего

принтера после печати документов.

**Входные и выходные данные**

Входной строкой является код на языке Kotlin, который описывает два класса: MainActivity и CrimeFragment. Класс MainActivity наследуется от AppCompatActivity и используется для создания главной активности приложения. Он содержит метод onCreate, который устанавливает макет для активности и добавляет фрагмент CrimeFragment, если он еще не создан.

Класс CrimeFragment наследуется от Fragment и используется для создания фрагмента, который позволяет пользователю вводить информацию о преступлении. Он содержит элементы управления, такие как EditText для ввода названия преступления, CheckBox для указания, решено ли преступление, и Button для выбора даты. Он также содержит методы onCreate и onCreateView, которые устанавливают начальное состояние фрагмента, и метод onStart, который регистрирует слушатели изменений для элементов управления и обновляет объект Crime при необходимости.

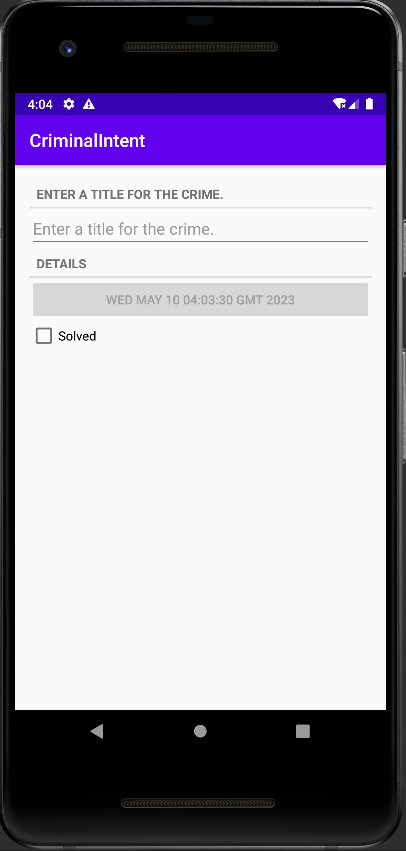
Выходными данными в данном случае является графический интерфейс пользователя, который позволяет ему вводить данные о преступлении и сохранять их в объекте Crime.

**Листинг программы (если есть)**

package com.bignerdranch.android.pr191  
  
import android.os.Bundle  
import android.support.v4.app.Fragment  
import android.text.Editable  
import android.text.TextWatcher  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.Button  
import android.widget.CheckBox  
import android.widget.EditText  
  
class CrimeFragment:Fragment() {  
 private lateinit var crime:Crime  
 private lateinit var titleField:EditText  
 private lateinit var dateButton:Button  
 private lateinit var solverCheckBox: CheckBox  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 crime=Crime()  
 }  
  
 override fun onCreateView(  
 inflater: LayoutInflater,  
 container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View? {  
 val view=inflater.inflate(R.layout.*fragment\_crime*,container,false)  
 titleField=view.findViewById(R.id.*crime\_title*) as EditText  
 dateButton=view.findViewById(R.id.*crime\_date*) as Button  
 solverCheckBox=view.findViewById(R.id.*crime\_solved*) as CheckBox  
 dateButton.*apply* **{** *text*=crime.date.toString()  
 *isEnabled*=false  
 **}** return view  
 }  
  
 override fun onStart() {  
 super.onStart()  
 val titleWatcher=object :TextWatcher{  
 override fun beforeTextChanged(  
 sequence: CharSequence?,  
 start: Int, count: Int, after: Int) {  
  
 }  
  
 override fun onTextChanged(  
 sequence: CharSequence?,  
 start: Int, before: Int, count: Int) {  
 crime.title=sequence.*toString*()  
 }  
  
 override fun afterTextChanged(sequence: Editable?) {  
  
 }  
 }  
 titleField.addTextChangedListener(titleWatcher)  
 solverCheckBox.*apply* **{** setOnCheckedChangeListener**{** \_,isChecked**->** crime.isSolved=isChecked  
 dateButton.*isEnabled*=true  
 **}  
 }** }  
}

class MainActivity : AppCompatActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
 val currentFragment=*supportFragmentManager*.findFragmentById(R.id.*fragment\_container*)  
 if(currentFragment==null){  
 val fragment=CrimeFragment()  
 *supportFragmentManager* .beginTransaction()  
 .add(R.id.*fragment\_container*,fragment)  
 .commit()  
 }  
 }  
}

**Скрин-шот экранов**

****

**Тестовые ситуации**

**Вывод**

Приложение «Список дел» работает успешно, в нем имеется:

- приветственный экран с авторизацией пользователя, проверкой

ввода на пустоту и сохранением информации о пользователе через механизм

SharedPreferences.

- содержать форму ввода данных

- экран для вывода сохраненной информации