Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт «**Практическая работа №4**»

Выполнила: Зиятдинова Алина Ленаровна

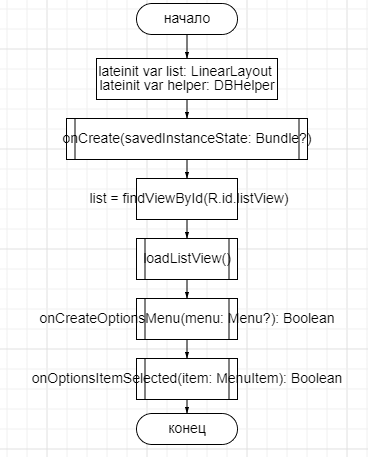
Группа: ПР-31

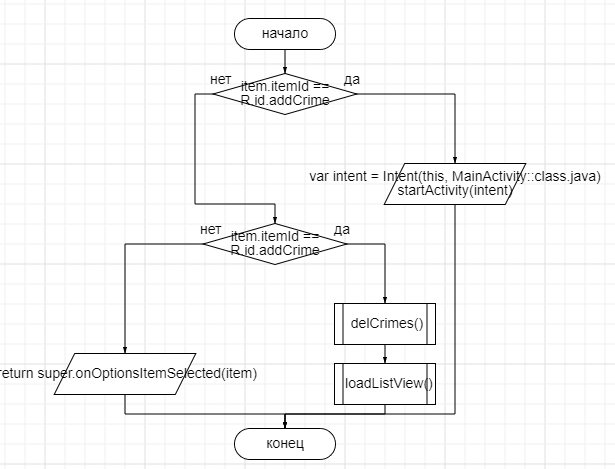
2023

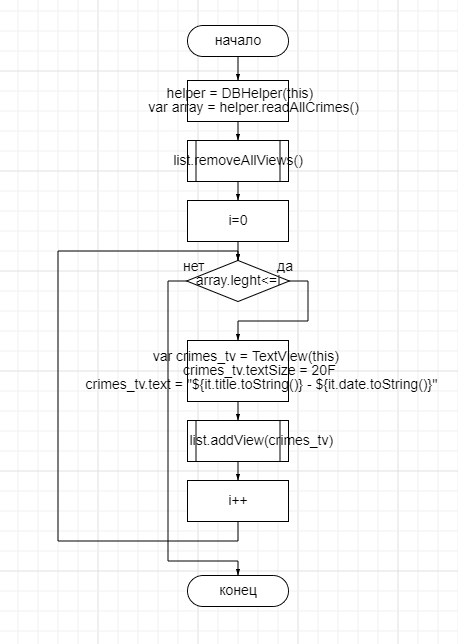
|  |
| --- |
| **build.gradle** |
| plugins **{** id 'com.android.application' version '7.3.1' apply false  id 'com.android.library' version '7.3.1' apply false  id 'org.jetbrains.kotlin.android' version '1.7.20' apply false **}**  plugins **{** id 'com.android.application'  id 'org.jetbrains.kotlin.android' **}** android **{** namespace 'com.example.prakt\_19\_1'  compileSdk 32   defaultConfig **{** applicationId "com.example.prakt\_19\_1"  minSdk 21  targetSdk 32  versionCode 1  versionName "1.0"   testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"  **}** buildTypes **{** release **{** minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  **}  }** compileOptions **{** sourceCompatibility JavaVersion.*VERSION\_1\_8* targetCompatibility JavaVersion.*VERSION\_1\_8* **}** kotlinOptions **{** jvmTarget = '1.8'  **}** viewBinding **{** enabled true  **} }** dependencies **{** implementation 'com.android.support:design:28.0.0'  implementation 'com.android.support:appcompat-v7:28.0.0'  implementation 'com.android.support.constraint:constraint-layout:2.0.4'  testImplementation 'junit:junit:4.13.2'  androidTestImplementation 'com.android.support.test:runner:1.0.2'  androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2' **}** |

|  |
| --- |
| **Manifest** |
| <?xml version="1.0" encoding="utf-8"?> <manifest xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools">   <application  android:allowBackup="true"  android:dataExtractionRules="@xml/data\_extraction\_rules"  android:fullBackupContent="@xml/backup\_rules"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:roundIcon="@mipmap/ic\_launcher\_round"  android:supportsRtl="true"  android:theme="@style/Theme.Prakt\_19\_1"  tools:targetApi="31">  <activity  android:name=".ListCrimeActivity"  android:exported="true">  <intent-filter>  <action android:name="android.intent.action.MAIN" />   <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  <meta-data  android:name="android.app.lib\_name"  android:value="" />  </activity>  <activity  android:name=".MainActivity"  android:exported="false">   <meta-data  android:name="android.app.lib\_name"  android:value="" />  </activity>  </application> </manifest> |

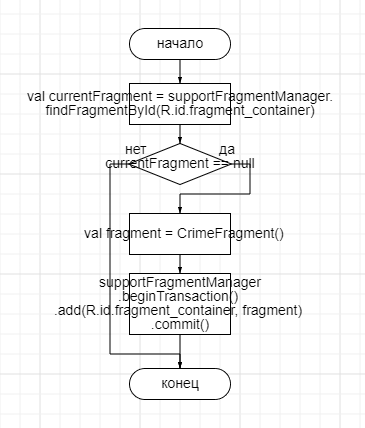
|  |
| --- |
| **ListCrimeActivity** |
| <?xml version="1.0" encoding="utf-8"?> <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical"  tools:context=".ListCrimeActivity">  <TextView style="?android:listSeparatorTextViewStyle"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="@string/allRecordsDB"/>   <LinearLayout  android:id="@+id/listView"  android:orientation="vertical"  android:layout\_width="match\_parent"  android:layout\_height="550dp">  </LinearLayout> </LinearLayout> |
| lateinit var *list*: LinearLayout lateinit var *helper*: DBHelper class ListCrimeActivity : AppCompatActivity() {  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.*activity\_list\_crime*)  *list* = findViewById(R.id.*listView*)  loadListView()  }  override fun onCreateOptionsMenu(menu: Menu?): Boolean {  val inflater = *menuInflater* inflater.inflate(R.menu.*menu\_top*, menu)  return super.onCreateOptionsMenu(menu)  }   override fun onOptionsItemSelected(item: MenuItem): Boolean {  when (item.*itemId*) {  R.id.*addCrime* -> {  var intent = Intent(this, MainActivity::class.*java*)  startActivity(intent)  return true  }  R.id.*delAll* -> {  delCrimes()  loadListView()  }  }  return super.onOptionsItemSelected(item)  }   fun delCrimes()  {  val db = DBHelper (this)  db.deleteAllCrimes()  loadListView()  }   fun loadListView()  {  *helper* = DBHelper(this)  var array = *helper*.readAllCrimes()  *list*.removeAllViews()  array.*forEach* **{** var crimes\_tv = TextView(this)  crimes\_tv.*textSize* = 20F  crimes\_tv.*text* = "${**it**.title.toString()} - ${**it**.date.toString()}"  *list*.addView(crimes\_tv)  **}** } } |

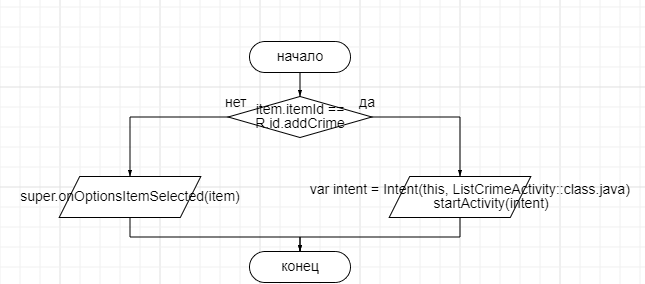






|  |
| --- |
| **MainActivity** |
| class MainActivity : AppCompatActivity() {  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.*activity\_main*)  val currentFragment = *supportFragmentManager*.findFragmentById(R.id.*fragment\_container*)  if (currentFragment == null)  {  val fragment = CrimeFragment()  *supportFragmentManager* .beginTransaction()  .add(R.id.*fragment\_container*, fragment)  .commit()  }  }  override fun onCreateOptionsMenu(menu: Menu?): Boolean {  val inflater = *menuInflater* inflater.inflate(R.menu.*menu\_top2*, menu)  return super.onCreateOptionsMenu(menu)  }   override fun onOptionsItemSelected(item: MenuItem): Boolean {  when (item.*itemId*) {  R.id.*show* -> {  var intent = Intent(this, ListCrimeActivity::class.*java*)  startActivity(intent)  return true  }  }  return super.onOptionsItemSelected(item)  } } |
| <FrameLayout  xmlns:android="http://schemas.android.com/apk/res/android"  android:id="@+id/fragment\_container"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  /> |





|  |
| --- |
| **Crime** |
| class Crime (var title:String, var date:String, var suspend: String) |

|  |
| --- |
| **DbContact** |
| object DBContract {   class CrimesEntry : BaseColumns {  companion object {  val TABLE\_NAME = "crimeDB"  val TITLE = "title"  var DATE = "date"  var SUSPEND = "suspend"  }  } } |

|  |
| --- |
| **DBHelper** |
| class DBHelper (val context: Context): SQLiteOpenHelper (context,"crimeDB", null, 1) {   override fun onCreate(db: SQLiteDatabase?) {  val quary = "CREATE TABLE crimeDB (id INT PRIMARY KEY, title TEXT, date TEXT, suspend TEXT)"  db!!.execSQL(quary)  }   override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {  db!!.execSQL("DROP TABLE IF EXISTS crimeDB")  onCreate(db)  }   fun addCrime(crime: Crime)  {  val values = ContentValues()  values.put("title", crime.title)  values.put("date", crime.date)  values.put("suspend", crime.suspend)  val db = this.*writableDatabase* db.insert("crimeDB", null, values)  db.close()  }   fun deleteAllCrimes() {  val db = *writableDatabase* db.delete("crimeDB", null, null)  }   @SuppressLint("Range")  fun readAllCrimes(): ArrayList<Crime> {  val crimes = ArrayList<Crime>()  val db = *writableDatabase* var cursor: Cursor? = null  try {  cursor = db.rawQuery("select \* from crimeDB", null)  } catch (e: SQLiteException) {  return ArrayList()  }  var title: String  var date: String  var suspend: String  if (cursor!!.moveToFirst()) {  while (cursor.*isAfterLast* == false) {  title = cursor.getString(cursor.getColumnIndex(DBContract.CrimesEntry.TITLE))  date = cursor.getString(cursor.getColumnIndex(DBContract.CrimesEntry.DATE))  suspend = cursor.getString(cursor.getColumnIndex(DBContract.CrimesEntry.SUSPEND))  crimes.add(Crime(title, date, suspend))  cursor.moveToNext()  }  }  return crimes  }   fun getCrime(title: String):Boolean //в функцию передается пароль и логин  {  val db = this.*readableDatabase* // поиск через запрос  val result = db.rawQuery("SELECT \* FROM crimeDB WHERE title = '$title'",null)  //выводим true или false  return result.moveToFirst()  } } |