**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное

учреждение высшего образования

**Национальный исследовательский**

**Томский политехнический университет**

Инженерная школа информационных технологий и робототехники

Отделение информационных технологий

Отчет по лабораторной работе №13 по дисциплине

**«Язык Kotlin и основы разработки»**

Намерения

****

Выполнил:

Студент группы 1А22 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.К. Кравцов

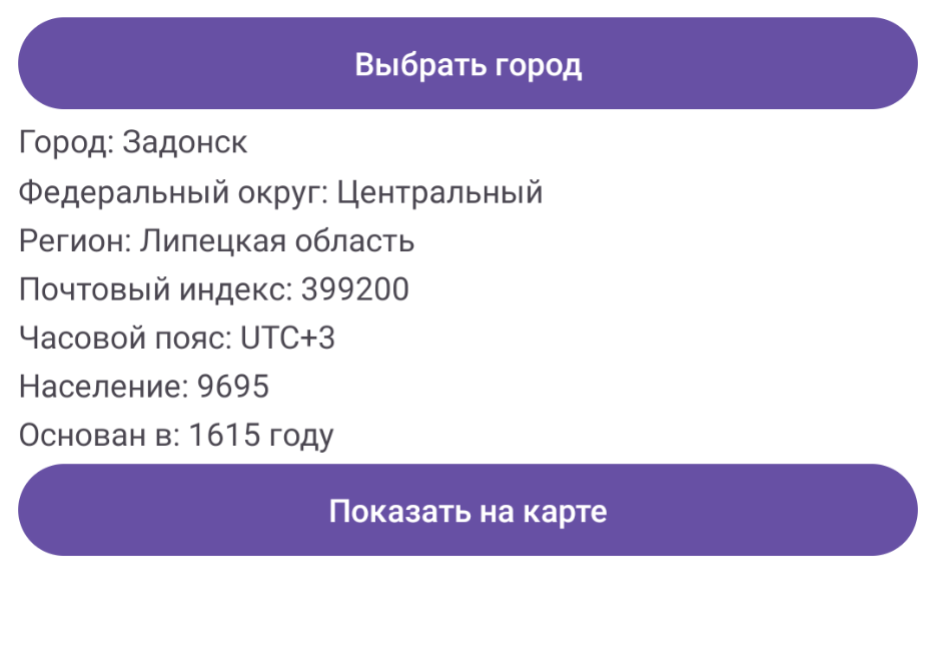
Проверил:

Ст. преп. ОИТ ИШИТР \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.А. Дорофеев

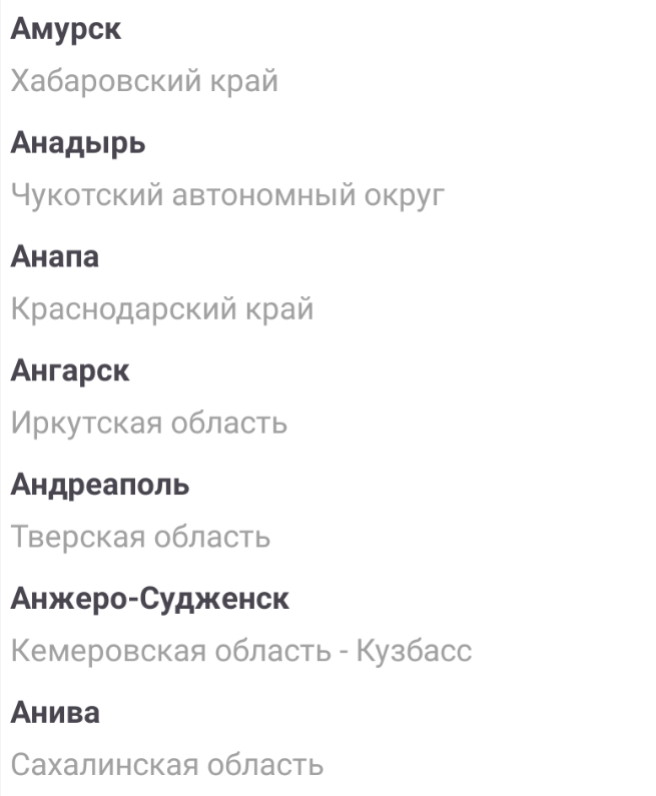
Томск 2025

# Задание

Создайте приложение, отображающее информацию о выбранном городе, а также показывающее его на карте. Внешний вид приложения должен быть примерно следующий:

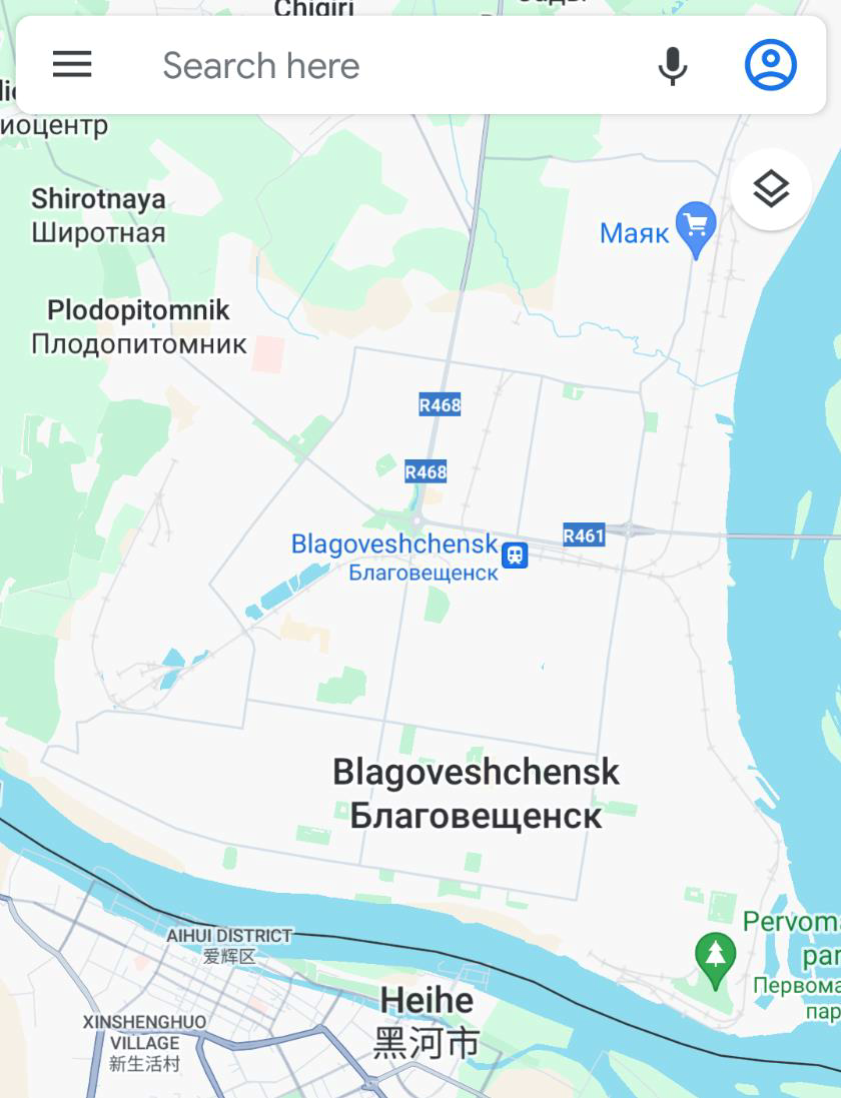


При нажатии кнопки «Выбрать город» запускается вторая активность, в которой отображается список RecyclerView с названиями городов и их регионов (ведь могут быть города с одинаковыми названиями, которые находятся в разных частях страны!). Вторая активность может выглядеть примерно так:



При нажатии на элемент списка активность закрывается, и информация о том какой город был выбран передаётся в основную активность, которая, в свою очередь, тут же отображает информацию об этом городе.

Кнопка «Показать на карте» запускает неявное намерение с координатами, система найдёт подходящее приложение для отображения карт и запустит его, показав нужный город:



Список городов с информацией о них можно найти в интернете, однако процесс её сбора может быть достаточно трудоёмким, поэтому можно воспользоваться готовым файлом, приложенным к этой странице. Файл содержит данные в формате CSV, когда каждая строчка содержит одну запись, а поля в ней разделены точкой с запятой или каким-то другим разделителем. В общем случае для работы с таким файлом крайне желательно использовать какую-либо библиотеку, поскольку часто данные могут содержать кавычки и их правильное разбиение может оказаться нетривиальным процессом. Однако в нашем файле данные довольно простые, поэтому используем встроенные функции для их чтения.

.

# Ход работы

1. Создан проект Lab13 на основе Empty Views Activity.
2. Подготовлены ресурсы приложения:

* Строковые ресурсы в strings.xml

<resources>  
 <string name="app\_name">Lab13</string>  
 <string name="choose\_city">Выбрать город</string>  
 <string name="show\_on\_map">Показать на карте</string>  
 <string name="city">Город: %s</string>  
 <string name="federal\_district">Федеральный округ: %s</string>  
 <string name="region">Регион: %s</string>  
 <string name="postal\_code">Почтовый индекс: %s</string>  
 <string name="timezone">Часовой пояс: %s</string>  
 <string name="population">Население: %s</string>  
 <string name="founded">Основан в: %s</string>  
 <string name="unknown">Неизвестно</string>  
 <string name="cities\_list\_title">Выберите город</string>  
 <string name="error\_loading\_cities">Ошибка загрузки городов</string>  
 <string name="error\_parsing\_data">Ошибка разбора данных</string>  
 <string name="geo\_uri\_template">geo:%1$.6f,%2$.6f?z=12</string>  
 <string name="browser\_map\_uri\_template">https://www.google.com/maps/@%1$.6f,%2$.6f,12z</string>  
</resources>

* Ресурсы размеров в dimens.xml

<?xml version="1.0" encoding="utf-8"?>  
<resources>  
 <dimen name="padding\_medium">16dp</dimen>  
 <dimen name="padding\_small">8dp</dimen>  
 <dimen name="padding\_very\_small">4dp</dimen>  
 <dimen name="margin\_bottom\_large">24dp</dimen>  
 <dimen name="margin\_top\_medium">24dp</dimen>  
 <dimen name="margin\_bottom\_medium">16dp</dimen>  
 <dimen name="text\_size\_large">18sp</dimen>  
 <dimen name="text\_size\_medium">16sp</dimen>  
 <dimen name="text\_size\_small">14sp</dimen>  
 <dimen name="item\_corner\_radius">8dp</dimen>  
 <dimen name="item\_elevation">2dp</dimen>  
</resources>

* Цветовые ресурсы в colors.xml

<?xml version="1.0" encoding="utf-8"?>  
<resources>  
 <color name="white">#FFFFFFFF</color>  
 <color name="primary\_color">#2196F3</color>  
 <color name="accent\_color">#FF4081</color>  
 <color name="background\_color">#FAFAFA</color>  
 <color name="card\_background">#FFFFFF</color>  
 <color name="text\_primary">#212121</color>  
 <color name="text\_secondary">#757575</color>  
 <color name="divider\_color">#E0E0E0</color>  
</resources>

* Файл cities.csv с данными о городах помещён в папку res/raw

1. Реализована модель данных City:

package ru.olegkravtsov.lab13  
  
data class City(  
 val title: String,  
 val region: String,  
 val district: String,  
 val postalCode: String,  
 val timezone: String,  
 val population: String,  
 val founded: String,  
 val lat: Float,  
 val lon: Float  
)

1. Создан синглтон Common для загрузки и хранения данных о городах:

package ru.olegkravtsov.lab13  
  
import android.content.Context  
import android.util.Log  
  
object Common {  
 val cities = mutableListOf<City>()  
  
 fun initCities(ctx: Context) {  
 if (cities.isEmpty()) {  
 try {  
 val inputStream = ctx.resources.openRawResource(R.raw.cities)  
 val lines = inputStream.bufferedReader().readLines()  
  
 for (i in 1 until lines.size) {  
 val line = lines[i].trim()  
 if (line.isNotEmpty()) {  
 val parts = line.split(";")  
 if (parts.size >= 9) {  
 try {  
 val city = City(  
 title = parts[3].trim(),  
 region = parts[2].trim(),  
 district = parts[1].trim(),  
 postalCode = parts[0].trim(),  
 timezone = parts[4].trim(),  
 population = parts[7].trim(),  
 founded = parts[8].trim(),  
 lat = parts[5].trim().toFloat(),  
 lon = parts[6].trim().toFloat()  
 )  
 cities.add(city)  
 } catch (e: NumberFormatException) {  
 Log.e("Common", ctx.getString(R.string.error\_parsing\_data), e)  
 } catch (e: Exception) {  
 Log.e("Common", ctx.getString(R.string.error\_parsing\_data), e)  
 }  
 }  
 }  
 }  
 cities.sortBy { it.title }  
 Log.d("Common", "Loaded ${cities.size} cities")  
 } catch (e: Exception) {  
 Log.e("Common", ctx.getString(R.string.error\_loading\_cities), e)  
 }  
 }  
 }  
}

1. Реализована основная активность MainActivity с явным и неявным намерениями:

package ru.olegkravtsov.lab13  
  
import android.content.Intent  
import android.net.Uri  
import android.os.Bundle  
import androidx.activity.result.contract.ActivityResultContracts  
import androidx.appcompat.app.AppCompatActivity  
import ru.olegkravtsov.lab13.databinding.ActivityMainBinding  
  
class MainActivity : AppCompatActivity() {  
  
 private lateinit var binding: ActivityMainBinding  
 private var selectedCity: City? = null  
  
 private val cityResultLauncher = registerForActivityResult(  
 ActivityResultContracts.StartActivityForResult()  
 ) { result ->  
 if (result.resultCode == RESULT\_OK) {  
 val data = result.data  
 val cityIndex = data?.getIntExtra("selected\_city\_index", -1) ?: -1  
 if (cityIndex != -1 && cityIndex < Common.cities.size) {  
 selectedCity = Common.cities[cityIndex]  
 updateCityInfo()  
 }  
 }  
 }  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 binding = ActivityMainBinding.inflate(layoutInflater)  
 setContentView(binding.root)  
  
 Common.initCities(this)  
  
 binding.btnChooseCity.setOnClickListener {  
 val intent = Intent(this, CityListActivity::class.java)  
 cityResultLauncher.launch(intent)  
 }  
  
 binding.btnShowOnMap.setOnClickListener {  
 selectedCity?.let { city ->  
 val geoUri = "geo:${city.lat},${city.lon}?z=12"  
 val mapIntent = Intent(Intent.ACTION\_VIEW, Uri.parse(geoUri))  
  
 try {  
 startActivity(mapIntent)  
 } catch (e: Exception) {  
 // Fallback  
 val browserUri = "https://www.google.com/maps/@${city.lat},${city.lon},12z"  
 val browserIntent = Intent(  
 Intent.ACTION\_VIEW,  
 Uri.parse(browserUri)  
 )  
 startActivity(browserIntent)  
 }  
 }  
 }  
  
 updateCityInfo()  
 }  
  
 private fun updateCityInfo() {  
 selectedCity?.let { city ->  
 binding.tvCity.text = getString(R.string.city, city.title)  
 binding.tvDistrict.text = getString(R.string.federal\_district, city.district)  
 binding.tvRegion.text = getString(R.string.region, city.region)  
 binding.tvPostalCode.text = getString(R.string.postal\_code, city.postalCode)  
 binding.tvTimezone.text = getString(R.string.timezone, city.timezone)  
 binding.tvPopulation.text = getString(R.string.population, city.population)  
 binding.tvFounded.text = getString(R.string.founded, city.founded)  
 } ?: run {  
 val unknown = getString(R.string.unknown)  
 binding.tvCity.text = getString(R.string.city, unknown)  
 binding.tvDistrict.text = getString(R.string.federal\_district, unknown)  
 binding.tvRegion.text = getString(R.string.region, unknown)  
 binding.tvPostalCode.text = getString(R.string.postal\_code, unknown)  
 binding.tvTimezone.text = getString(R.string.timezone, unknown)  
 binding.tvPopulation.text = getString(R.string.population, unknown)  
 binding.tvFounded.text = getString(R.string.founded, unknown)  
 }  
 }  
}

1. Реализована активность со списком городов CityListActivity с RecyclerView:

package ru.olegkravtsov.lab13  
  
import android.content.Intent  
import android.os.Bundle  
import androidx.appcompat.app.AppCompatActivity  
import androidx.recyclerview.widget.LinearLayoutManager  
import ru.olegkravtsov.lab13.databinding.ActivityCityListBinding  
  
class CityListActivity : AppCompatActivity() {  
  
 private lateinit var binding: ActivityCityListBinding  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 binding = ActivityCityListBinding.inflate(layoutInflater)  
 setContentView(binding.root)  
  
 setSupportActionBar(binding.toolbar)  
 supportActionBar?.setDisplayHomeAsUpEnabled(true)  
 binding.toolbar.setNavigationOnClickListener {  
 finish()  
 }  
  
 val adapter = CityAdapter(Common.cities) { position ->  
 val resultIntent = Intent().apply {  
 putExtra("selected\_city\_index", position)  
 }  
 setResult(RESULT\_OK, resultIntent)  
 finish()  
 }  
  
 binding.rvCities.layoutManager = LinearLayoutManager(this)  
 binding.rvCities.adapter = adapter  
 }  
}

1. Создан адаптер для RecyclerView:

package ru.olegkravtsov.lab13  
  
import android.view.LayoutInflater  
import android.view.ViewGroup  
import androidx.recyclerview.widget.RecyclerView  
import ru.olegkravtsov.lab13.databinding.ItemCityBinding  
  
class CityAdapter(  
 private val cities: List<City>,  
 private val onItemClick: (Int) -> Unit  
) : RecyclerView.Adapter<CityAdapter.CityViewHolder>() {  
  
 class CityViewHolder(private val binding: ItemCityBinding) : RecyclerView.ViewHolder(binding.root) {  
 fun bind(city: City) {  
 binding.tvCityName.text = city.title  
 binding.tvRegion.text = city.region  
 }  
 }  
  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): CityViewHolder {  
 val binding = ItemCityBinding.inflate(LayoutInflater.from(parent.context), parent, false)  
 return CityViewHolder(binding)  
 }  
  
 override fun onBindViewHolder(holder: CityViewHolder, position: Int) {  
 val city = cities[position]  
 holder.bind(city)  
  
 holder.itemView.setOnClickListener {  
 onItemClick(position)  
 }  
 }  
  
 override fun getItemCount(): Int = cities.size  
}

**Результат работы**

Приложение успешно запускается и предоставляет интерфейс для выбора города (рис. 1) и просмотра информации о нем (рис. 2). Все функции работают согласно требованиям задания.

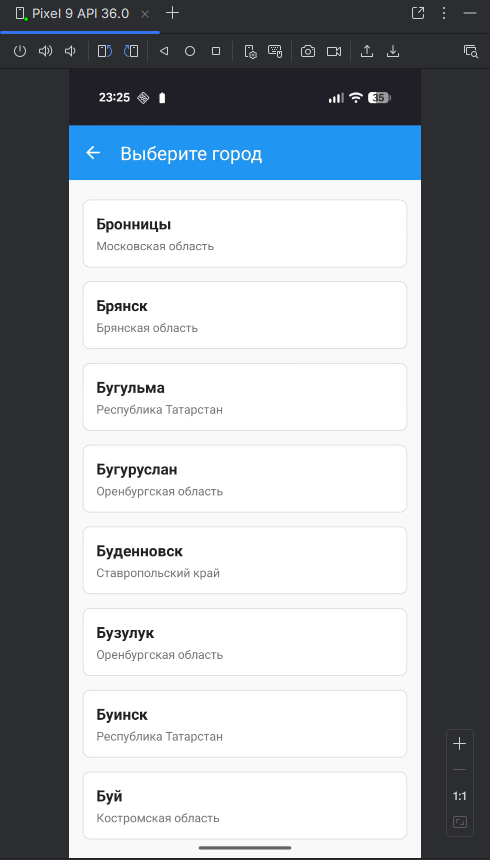


Рисунок 1 - Результат работы приложения (выбор города)

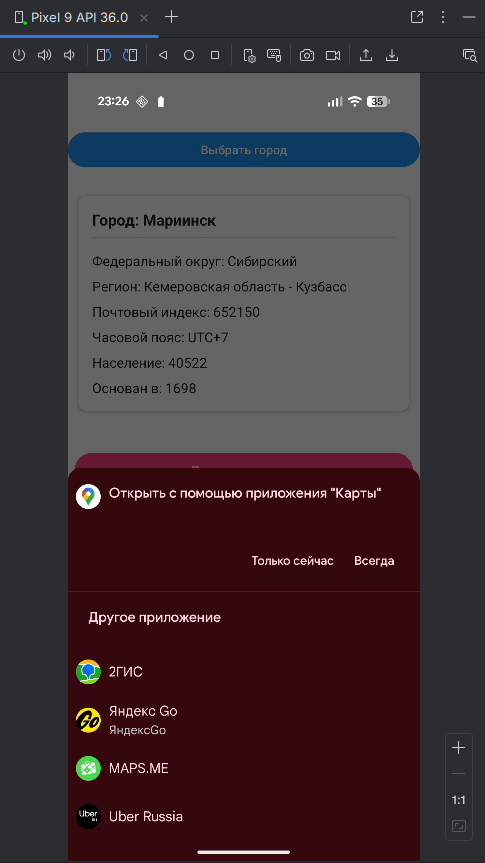
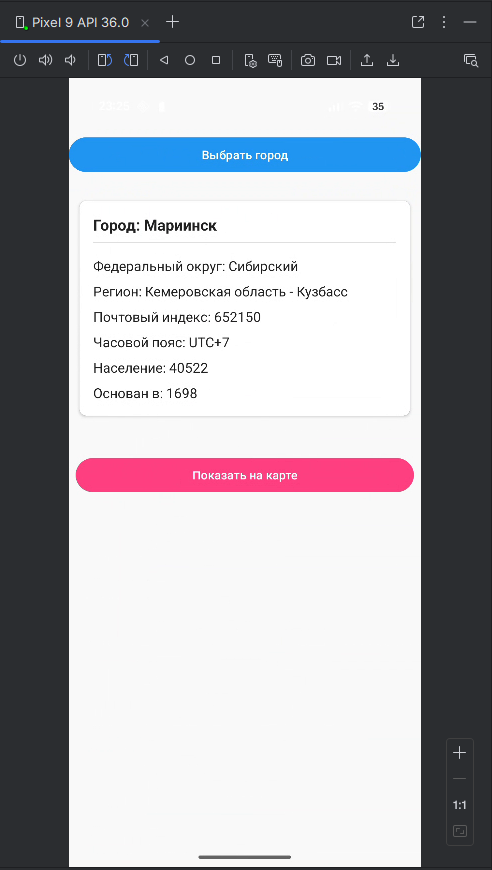


Рисунок 2 - Результат работы приложения (информация о городе)

# Выводы

В ходе выполнения лабораторной работы №13 были успешно освоены принципы работы с намерениями в Android-приложениях. Разработанное приложение демонстрирует практическое применение как явных, так и неявных намерений:

* Освоено использование явных намерений для запуска активности внутри приложения с возвратом результата через современный подход с registerForActivityResult
* Реализовано неявное намерение для отображения местоположения города в приложениях карт с обработкой возможных исключений
* Освоена работа с raw-ресурсами для загрузки данных из CSV-файла
* Применены навыки работы с RecyclerView для отображения списка городов
* Реализована архитектура приложения с использованием синглтона для хранения общих данных

Полученные навыки позволяют создавать приложения с многооконным интерфейсом и интеграцией с системными приложениями через механизм намерений.