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 №12 по дисциплине

«Язык Kotlin и основы разработки»

Элемент управления RecyclerView

Выполнил:

Студент группы 1А22  О.К. Кравцов

Проверил:

Ст. преп. ОИТ ИШИТР _____ В.А. Дорофеев

Томск 2025

Задание

Создайте приложение, показывающее пользователю витрину онлайн-магазина. Приложение должно выглядеть примерно следующим образом:

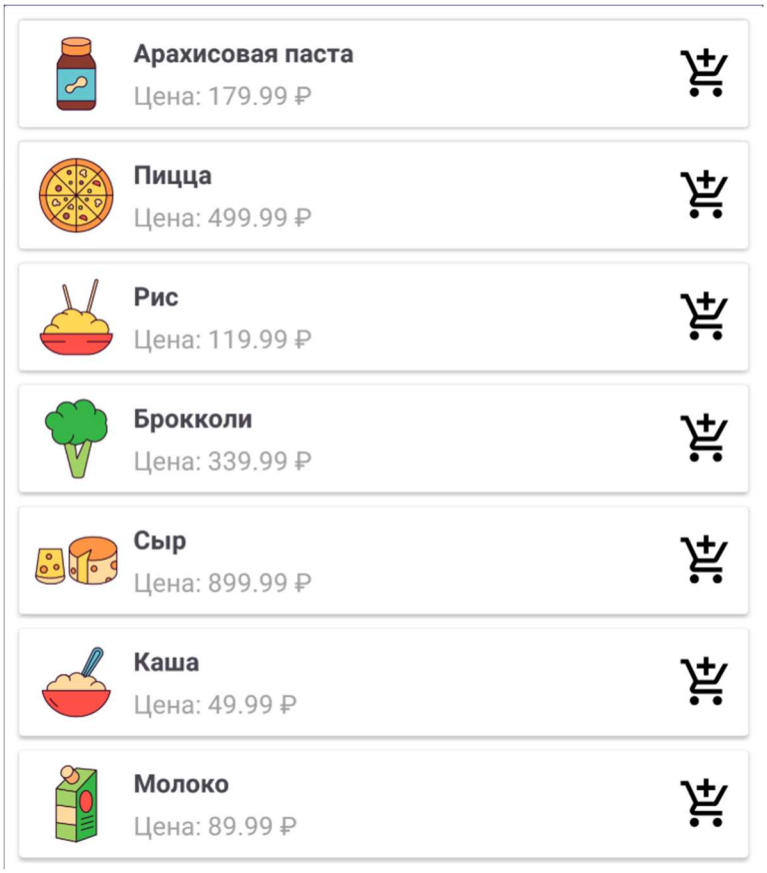

Рисунок 1 – Пример интерфейса

Список должен быть организован с помощью элемента RecyclerView.

Разметка элемента списка должна быть сделана на основе CardView. Он представляет собой контейнер на базе FrameLayout, только добавляет рамку с возможностью скругления углов, тень и т.д. Внутрь можно поместить другой контейнер, ведь FrameLayout и его наследники – не самые удобные в плане размещения элементов.

Каждый элемент должен содержать изображение товара, название, стоимость, а также кнопку добавления товара в тележку. В примере выше изображения товаров взяты из лабораторной работы 8 (игра «Съедобно-несъедобно»), но можно взять любые другие товары.

При нажатии кнопки со значком добавления товара в тележку информация об этом должна передаваться в основную активность:

– Если товар ещё не был добавлен, то появляется всплывающее сообщение зелёного цвета с текстом «Добавлено: <название товара>», а значок меняется на тележку без знака «+».

– Если товар уже был добавлен, то появляется всплывающее сообщение красного цвета с текстом «Удалено: <название товара>», а значок меняется обратно на тележку со знаком «+».

Значки тележки со знаком «+» и без него приложены в архиве, однако можно использовать другие аналогичные по смыслу значки. Чтобы добавить векторное изображение (SVG-файл со значком) в ресурсы, нужно щёлкнуть правой кнопкой по папке res → drawable, и в меню выбрать New → Vector Asset, а затем в появившемся окне выбрать SVG-файл и добавить его.

.

Ход работы

1. Создан проект Lab12 на основе Empty Views Activity.
2. Подготовлены ресурсы приложения:

– Строковые ресурсы в strings.xml

```
<resources>
    <string name="app_name">Lab12</string>
    <string name="removed">Удалено: %s</string>
    <string name="added">Добавлено: %s</string>

    <!-- Названия товаров -->
    <string name="apple_juice">Яблочный сок</string>
    <string name="broccoli">Брокколи</string>
    <string name="carrot">Морковь</string>
    <string name="cheese">Сыр</string>
    <string name="chicken_drumstick">Куриная ножка</string>
    <string name="crab">Краб</string>
    <string name="donut">Донат</string>
    <string name="egg">Яйцо</string>
    <string name="fish">Форель</string>
    <string name="french_fries_basket">Картошка фри</string>
    <string name="ice_cream">Мороженое</string>
    <string name="milk">Молоко</string>
    <string name="mushroom">Грибы</string>
    <string name="noodles">Лапша</string>
    <string name="oatmeal">Каша овсяная</string>
    <string name="pancakes">Блинчики</string>
    <string name="peanut_butter">Арахисовая паста</string>
    <string name="pizza">Пицца</string>
    <string name="sliced_bread">Хлеб</string>
    <string name="sushi_rolls">Суши</string>

    <!-- Цены -->
    <string name="apple_juice_price">195,99 руб</string>
    <string name="broccoli_price">379,99 руб/кг</string>
    <string name="carrot_price">22,99 руб/кг</string>
    <string name="cheese_price">1249,99 руб/кг</string>
    <string name="chicken_drumstick_price">850,99 руб/кг</string>
    <string name="crab_price">7600,99 руб/кг</string>
    <string name="donut_price">49,99 руб</string>
    <string name="egg_price">109,99 руб/10 шт</string>
    <string name="fish_price">2199,99 руб/кг</string>
    <string name="french_fries_basket_price">239,99 руб</string>
    <string name="ice_cream_price">109,99 руб</string>
    <string name="milk_price">89,99 руб</string>
    <string name="mushroom_price">619,99 руб/кг</string>
    <string name="noodles_price">74,99 руб</string>
    <string name="oatmeal_price">135,99 руб</string>
    <string name="pancakes_price">299,99 руб</string>
    <string name="peanut_butter_price">489,99 руб</string>
    <string name="pizza_price">577,99 руб</string>
    <string name="sliced_bread_price">55,99 руб</string>
    <string name="sushi_rolls_price">199,99 руб</string>
</resources>
```

– Ресурсы размеров в dimens.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <!-- Отступы -->
    <dimen name="card_margin">8dp</dimen>
    <dimen name="card_corner_radius">8dp</dimen>
    <dimen name="card_elevation">4dp</dimen>
    <dimen name="card_padding">12dp</dimen>
    <dimen name="text_margin_top">8dp</dimen>
    <dimen name="text_small_margin_top">4dp</dimen>

    <!-- Размеры -->
    <dimen name="product_image_height">150dp</dimen>
    <dimen name="cart_button_size">40dp</dimen>
    <dimen name="product_name_text_size">16sp</dimen>
    <dimen name="product_price_text_size">14sp</dimen>
</resources>
```

– Цветовые ресурсы в colors.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <!-- Цвета для сообщений Snackbar -->
    <color name="snackbar_success">#4CAF50</color>
    <color name="snackbar_error">#F44336</color>

    <!-- Цвета текста для лучшей читаемости -->
    <color name="snackbar_success_text">#FFFFFF</color>
    <color name="snackbar_error_text">#FFFFFF</color>
</resources>
```

– Растровые изображения товаров (в папке res/drawable (apple_juice.png, broccoli.png, carrot.png и др.))

– Векторные иконки для кнопки корзины (shopping_cart.svg и shopping_cart_add.svg) были импортированы через Vector Asset Studio

3. Создан класс Product для хранения данных о товаре:

```kotlin
package ru.olegkravtsov.lab12

data class Product(
    val id: Long,
    val name: String,
    val price: String,
    val imageResId: Int
)
```

4. Реализована разметка элемента списка item_product.xml на основе CardView:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="@dimen/card_margin"
    app:cardCornerRadius="@dimen/card_corner_radius"
    app:cardElevation="@dimen/card_elevation">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:padding="@dimen/card_padding">

        <ImageView
            android:id="@+id/productImage"
            android:layout_width="match_parent"
            android:layout_height="@dimen/product_image_height"
            android:scaleType="centerCrop" />

        <TextView
            android:id="@+id/productName"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/text_margin_top"
            android:textSize="@dimen/product_name_text_size"
            android:textStyle="bold" />

        <TextView
            android:id="@+id/productPrice"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/text_small_margin_top"
            android:textSize="@dimen/product_price_text_size" />

        <ImageButton
            android:id="@+id/cartButton"
            android:layout_width="@dimen/cart_button_size"
            android:layout_height="@dimen/cart_button_size"
            android:layout_gravity="end"
            android:background="?attr/selectableItemBackgroundBorderless"
            android:scaleType="centerInside" />

    </LinearLayout>

</androidx.cardview.widget.CardView>
```

5. Реализован адаптер ProductAdapter для RecyclerView:

```
package ru.olegkravtsov.lab12

import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.core.content.ContextCompat
import androidx.recyclerview.widget.RecyclerView
import com.google.android.material.snackbar.Snackbar
import ru.olegkravtsov.lab12.databinding.ItemProductBinding

class ProductAdapter(
    private val products: List<Product>,
    private val rootView: View
) : RecyclerView.Adapter<ProductAdapter.ProductViewHolder>() {

    private val cartItems = mutableSetOf<Long>()

    class ProductViewHolder(private val binding: ItemProductBinding) :
        RecyclerView.ViewHolder(binding.root) {

        fun bind(product: Product, isInCart: Boolean, onCartClick: (Product) -> Unit)
{
            binding.productImage.setImageResource(product.imageResId)
            binding.productName.text = product.name
            binding.productPrice.text = product.price

            val iconRes = if (isInCart) {
                R.drawable.shopping_cart
            } else {
                R.drawable.shopping_cart_add
            }
            binding.cartButton.setImageResource(iconRes)

            binding.cartButton.setOnClickListener {
                onCartClick(product)
            }
        }
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
ProductViewHolder {
        val binding = ItemProductBinding.inflate(
            LayoutInflater.from(parent.context),
            parent,
            false
        )
        return ProductViewHolder(binding)
    }

    override fun onBindViewHolder(holder: ProductViewHolder, position: Int) {
        val product = products[position]
        val isInCart = cartItems.contains(product.id)
        holder.bind(product, isInCart) { clickedProduct ->
            handleCartClick(clickedProduct, holder)
        }
```

```
    }

    override fun getItemCount(): Int = products.size

    private fun handleCartClick(product: Product, holder: ProductViewHolder) {
        val wasInCart = cartItems.contains(product.id)

        if (wasInCart) {
            cartItems.remove(product.id)
            showSnackbar(
                rootView.context.getString(R.string.removed, product.name),
                ContextCompat.getColor(rootView.context, R.color.snackbar_error)
            )
        } else {
            cartItems.add(product.id)
            showSnackbar(
                rootView.context.getString(R.string.added, product.name),
                ContextCompat.getColor(rootView.context, R.color.snackbar_success)
            )
        }

        notifyItemChanged(holder.bindingAdapterPosition)
    }

    private fun showSnackbar(message: String, backgroundColor: Int) {
        val snackbar = Snackbar.make(rootView, message, Snackbar.LENGTH_SHORT)

        // Используем методы из лабораторной работы №10 для настройки цвета
        snackbar.setBackgroundTint(backgroundColor)

        // Устанавливаем цвет текста для лучшей читаемости
        val textColor = if (backgroundColor ==
ContextCompat.getColor(rootView.context, R.color.snackbar_success)) {
            ContextCompat.getColor(rootView.context, R.color.snackbar_success_text)
        } else {
            ContextCompat.getColor(rootView.context, R.color.snackbar_error_text)
        }
        snackbar.setTextColor(textColor)

        snackbar.show()
    }
}
```

6. Реализована основная активность MainActivity.kt с настройкой RecyclerView:

```
package ru.olegkravtsov.lab12

import android.os.Bundle
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat
import androidx.recyclerview.widget.GridLayoutManager
import ru.olegkravtsov.lab12.databinding.ActivityMainBinding

class MainActivity : AppCompatActivity() {
    private lateinit var binding: ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()

        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)

        ViewCompat.setOnApplyWindowInsetsListener(binding.main) { v, insets ->
            val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom)
            insets
        }

        setupRecyclerView()
    }

    private fun setupRecyclerView() {
        val products = listOf(
            Product(1, getString(R.string.apple_juice),
getString(R.string.apple_juice_price), R.drawable.apple_juice),
            Product(2, getString(R.string.broccoli),
getString(R.string.broccoli_price), R.drawable.broccoli),
            Product(3, getString(R.string.carrot), getString(R.string.carrot_price),
R.drawable.carrot),
            Product(4, getString(R.string.cheese), getString(R.string.cheese_price),
R.drawable.cheese),
            Product(5, getString(R.string.chicken_drumstick),
getString(R.string.chicken_drumstick_price), R.drawable.chicken_drumstick),
            Product(6, getString(R.string.crab), getString(R.string.crab_price),
R.drawable.crab),
            Product(7, getString(R.string.donut), getString(R.string.donut_price),
R.drawable.donut),
            Product(8, getString(R.string.egg), getString(R.string.egg_price),
R.drawable.egg),
            Product(9, getString(R.string.fish), getString(R.string.fish_price),
R.drawable.fish),
            Product(10, getString(R.string.french_fries_basket),
getString(R.string.french_fries_basket_price), R.drawable.french_fries_basket),
            Product(11, getString(R.string.ice_cream),
getString(R.string.ice_cream_price), R.drawable.ice_cream),
            Product(12, getString(R.string.milk), getString(R.string.milk_price),
```

```
R.drawable.milk),
            Product(13, getString(R.string.mushroom),
getString(R.string.mushroom_price), R.drawable.mushroom),
            Product(14, getString(R.string.noodles),
getString(R.string.noodles_price), R.drawable.noodles),
            Product(15, getString(R.string.oatmeal),
getString(R.string.oatmeal_price), R.drawable.oatmeal),
            Product(16, getString(R.string.pancakes),
getString(R.string.pancakes_price), R.drawable.pancakes),
            Product(17, getString(R.string.peanut_butter),
getString(R.string.peanut_butter_price), R.drawable.peanut_butter),
            Product(18, getString(R.string.pizza), getString(R.string.pizza_price),
R.drawable.pizza),
            Product(19, getString(R.string.sliced_bread),
getString(R.string.sliced_bread_price), R.drawable.sliced_bread),
            Product(20, getString(R.string.sushi_rolls),
getString(R.string.sushi_rolls_price), R.drawable.sushi_rolls)
        )

        val adapter = ProductAdapter(products, binding.root)
        binding.productsRecyclerView.layoutManager = GridLayoutManager(this, 2)
        binding.productsRecyclerView.adapter = adapter
    }
}
```

Результат работы

Приложение успешно запускается и отображает витрину товаров в виде сетки (2 колонки). Каждый товар представлен в виде карточки CardView с растровым изображением товара, названием, ценой и кнопкой добавления в корзину (рис. 2).

При нажатии на кнопку корзины:

– Если товар не в корзине - иконка меняется на тележку, появляется зелёное сообщение "Добавлено: <название>"

– Если товар в корзине - иконка меняется на тележку с плюсом, появляется красное сообщение "Удалено: <название>"

Все сообщения Snackbar используют цветовое оформление согласно требованиям задания.

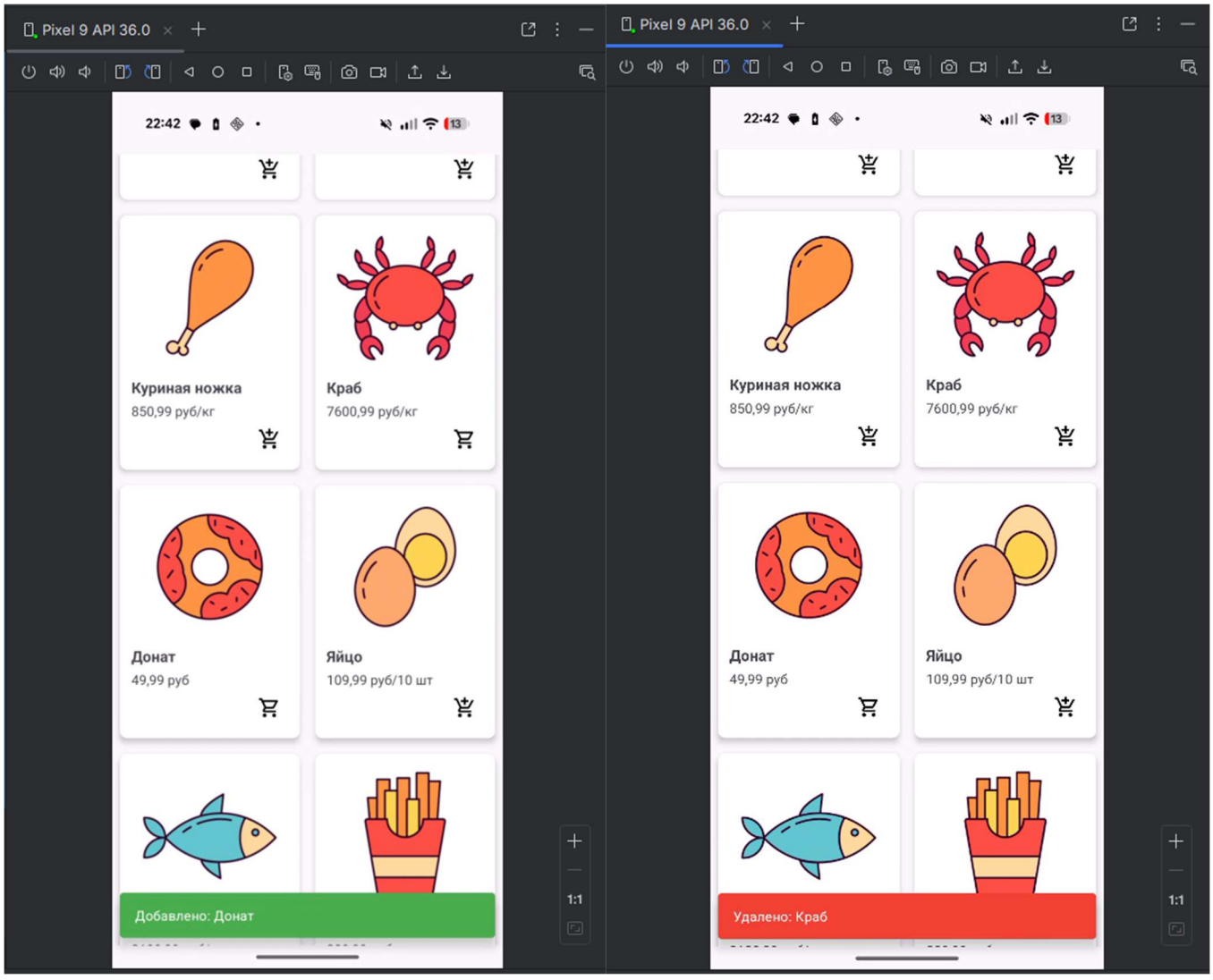

Рисунок 2 - Результат работы приложения

Выводы

В ходе выполнения лабораторной работы было успешно разработано приложение для отображения витрины онлайн-магазина с использованием элемента управления RecyclerView. Освоена работа с компонентом RecyclerView, включая создание адаптера, ViewHolder и настройку менеджера разметки GridLayoutManager.

Добавлены и использованы растровые изображения товаров в ресурсах приложения, что обеспечило наглядное и привлекательное отображение товаров в витрине. Реализовано динамическое изменение состояния элементов списка при взаимодействии пользователя с кнопкой добавления в корзину.

Применён компонент CardView для создания визуально привлекательных карточек товаров с скруглёнными углами и тенями. Использованы наработки из лабораторной работы №10 для отображения стилизованных сообщений Snackbar с различными цветами фона в зависимости от типа операции.

Все функции работают согласно требованиям задания, обеспечивая интуитивно понятный и отзывчивый интерфейс для пользователя. Полученные навыки позволяют создавать сложные списки и сетки элементов с кастомным оформлением и интерактивными возможностями.