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 №16 по дисциплине

«Язык Kotlin и основы разработки»

Навигация

Выполнил:

Студент группы 1А22 _____ О.К. Кравцов

Проверил:

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

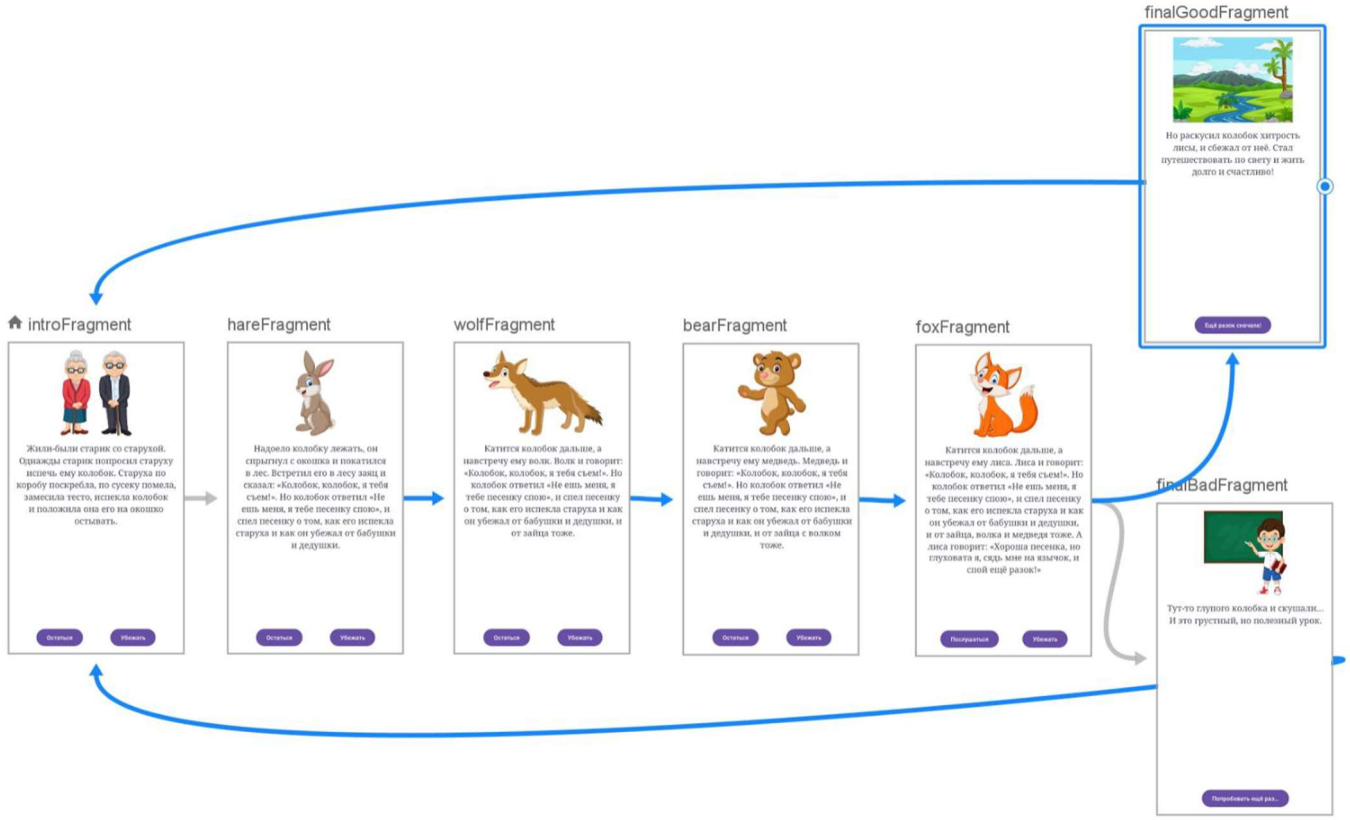
Томск 2025

Задание

Разработать приложение с интерактивной сказкой «Колобок». Приложение состоит из ряда фрагментов, отображающих иллюстрированные части сказки. Переходы между фрагментами осуществляются по нажатию кнопок в нижней части экрана. Иллюстрации можно взять из приложенного архива.

Вместо сказки «Колобок» по желанию можно использовать какую-либо другую историю, но механизм приложения должен остаться примерно таким же.

Навигационный граф для приложения может выглядеть примерно таким образом:



При желании допускается использовать один и тот же фрагмент несколько раз, передавая ему в качестве параметров информацию о том какая часть истории должна отображаться.

Ход работы

1. Создан проект Lab16 на основе Empty Views Activity.
2. Настроены ресурсы приложения:

– strings.xml

```
<resources>
    <string name="app_name">Lab16</string>
    <string name="story_image">Иллюстрация к сказке</string>

    <!-- Тексты фрагментов -->
    <string name="intro_story">Жили-были старик со старухой. Однажды старик попросил старуху испечь ему колобок. Старуха по коробу поскребла, по сусеку помела, замесила тесто, испекла колобок и положила она его на окошко остывать.</string>
    <string name="hare_story">Надоело колобку лежать, он спрыгнул с окошка и покатился в лес. Встретил его в лесу заяц и сказал: «Колобок, колобок, я тебя съем!». Но колобок ответил «Не ешь меня, я тебе песенку спою», и спел песенку о том, как его испекла старуха и как он убежал от бабушки и дедушки.</string>
    <string name="wolf_story">Катится колобок дальше, а навстречу ему волк. Волк и говорит: «Колобок, колобок, я тебя съем!». Но колобок ответил «Не ешь меня, я тебе песенку спою», и спел песенку о том, как его испекла старуха и как он убежал от бабушки и дедушки, и от зайца тоже.</string>
    <string name="bear_story">Катится колобок дальше, а навстречу ему медведь. Медведь и говорит: «Колобок, колобок, я тебя съем!». Но колобок ответил «Не ешь меня, я тебе песенку спою», и спел песенку о том, как его испекла старуха и как он убежал от бабушки и дедушки, и от зайца с волком тоже.</string>
    <string name="fox_story">Катится колобок дальше, а навстречу ему лиса. Лиса и говорит: «Колобок, колобок, я тебя съем!». Но колобок ответил «Не ешь меня, я тебе песенку спою», и спел песенку о том, как его испекла старуха и как он убежал от бабушки и дедушки, и от зайца, волка и медведя тоже. А лиса говорит: «Хороша песенка, но глуховата я, сядь мне на язычок, и спой ещё разок!»</string>
    <string name="final_bad_story">Тут-то глупого колобка и скушали... И это грустный, но полезный урок.</string>
    <string name="final_good_story">Но раскусил колобок хитрость лисы, и сбежал от неё. Стал путешествовать по свету и жить долго и счастливо!</string>

    <!-- Варианты выбора -->
    <string name="choice_run">Убежать</string>
    <string name="choice_stay">Остаться</string>
    <string name="choice_obey">Послушаться</string>
    <string name="choice_retry">Попробовать ещё раз…</string>
    <string name="choice_restart">Ещё разок сначала!</string>
</resources>
```

– colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
    <color name="primary_color">#2196F3</color>
</resources>
```

– dimens.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <dimen name="default_padding">16dp</dimen>
    <dimen name="button_bottom_margin">8dp</dimen>
    <dimen name="element_spacing">16dp</dimen>
    <dimen name="text_size_medium">18sp</dimen>
    <dimen name="line_spacing">4sp</dimen>
    <dimen name="image_max_height">400dp</dimen>
    <dimen name="button_padding">16dp</dimen>
    <dimen name="status_bar_padding">32dp</dimen>
</resources>
```

– themes.xml

```
<resources>
    <style name="Base.Theme.Lab16"
parent="Theme.Material3.Light.NoActionBar">
        <item name="android:windowBackground">@color/white</item>
        <item name="colorPrimary">@color/primary_color</item>
        <item name="colorOnPrimary">@color/white</item>
        <item name="android:colorBackground">@color/white</item>
        <item name="android:statusBarColor">@android:color/transparent</item>
        <item name="android:windowDrawsSystemBarBackgrounds">true</item>
        <item name="android:windowLightStatusBar">true</item>
    </style>

    <style name="Theme.Lab16" parent="Base.Theme.Lab16" />
</resources>
```

– В папку res/drawable добавлены иллюстрации для всех сцен сказки: elders, hare, wolf, bear, fox, final_bad, final_good.

3. Реализован абстрактный класс BaseStoryFragment, содержащий общую логику для всех фрагментов приложения:

```
package ru.olegkravtsov.lab16

import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.Button
import android.widget.ImageView
import android.widget.LinearLayout
import android.widget.TextView
import androidx.core.content.ContextCompat
import androidx.fragment.app.Fragment

abstract class BaseStoryFragment : Fragment() {

    abstract val imageRes: Int
    abstract val storyTextRes: Int
    abstract val choices: List<Choice>

    data class Choice(
        val textRes: Int,
        val action: () -> Unit
    )

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        return inflater.inflate(R.layout.fragment_story, container, false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)


view.findViewById<ImageView>(R.id.storyImage).setImageResource(imageRes)
        view.findViewById<TextView>(R.id.storyText).text =
getString(storyTextRes)

        val buttonsContainer =
view.findViewById<LinearLayout>(R.id.buttonsContainer)
        buttonsContainer.removeAllViews()

        choices.forEach { choice ->
            val button = Button(requireContext()).apply {
                text = getString(choice.textRes)
                layoutParams = LinearLayout.LayoutParams(
```

```
                    LinearLayout.LayoutParams.MATCH_PARENT,
                    LinearLayout.LayoutParams.WRAP_CONTENT
                ).apply {
                    bottomMargin =
resources.getDimensionPixelSize(R.dimen.button_bottom_margin)
                }

                setBackgroundColor(ContextCompat.getColor(requireContext(),
R.color.primary_color))
                setTextColor(ContextCompat.getColor(requireContext(),
R.color.white))
                textSize = 16f
                setPadding(
                    resources.getDimensionPixelSize(R.dimen.button_padding),
                    resources.getDimensionPixelSize(R.dimen.button_padding),
                    resources.getDimensionPixelSize(R.dimen.button_padding),
                    resources.getDimensionPixelSize(R.dimen.button_padding)
                )

                setOnClickListener { choice.action() }
            }
            buttonsContainer.addView(button)
        }
    }
}
```

4. Для каждой сцены сказки создан соответствующий фрагмент, наследующий от BaseStoryFragment:

– IntroFragment - начальная сцена

```
package ru.olegkravtsov.lab16

import androidx.navigation.fragment.findNavController

class IntroFragment : BaseStoryFragment() {
    override val imageRes = R.drawable.elders
    override val storyTextRes = R.string.intro_story

    override val choices = listOf(
        Choice(R.string.choice_run) {

findNavController().navigate(R.id.action_introFragment_to_hareFragment)
        },
        Choice(R.string.choice_stay) {

findNavController().navigate(R.id.action_introFragment_to_finalBadFragment)
        }
    )
}
```

– HareFragment - встреча с зайцем

```
package ru.olegkravtsov.lab16

import androidx.navigation.fragment.findNavController

class HareFragment : BaseStoryFragment() {
    override val imageRes = R.drawable.hare
    override val storyTextRes = R.string.hare_story

    override val choices = listOf(
        Choice(R.string.choice_run) {

findNavController().navigate(R.id.action_hareFragment_to_wolfFragment)
        },
        Choice(R.string.choice_stay) {

findNavController().navigate(R.id.action_hareFragment_to_finalBadFragment)
        }
    )
}
```

– WolfFragment - встреча с волком

```
package ru.olegkravtsov.lab16

import androidx.navigation.fragment.findNavController

class WolfFragment : BaseStoryFragment() {
    override val imageRes = R.drawable.wolf
    override val storyTextRes = R.string.wolf_story

    override val choices = listOf(
        Choice(R.string.choice_run) {

findNavController().navigate(R.id.action_wolfFragment_to_bearFragment)
        },
        Choice(R.string.choice_stay) {

findNavController().navigate(R.id.action_wolfFragment_to_finalBadFragment)
        }
    )
}
```

– BearFragment - встреча с медведем

```
package ru.olegkravtsov.lab16

import androidx.navigation.fragment.findNavController

class BearFragment : BaseStoryFragment() {
    override val imageRes = R.drawable.bear
    override val storyTextRes = R.string.bear_story

    override val choices = listOf(
        Choice(R.string.choice_run) {

findNavController().navigate(R.id.action_bearFragment_to_foxFragment)
        },
        Choice(R.string.choice_stay) {

findNavController().navigate(R.id.action_bearFragment_to_finalBadFragment)
        }
    )
}
```

– FoxFragment - встреча с лисой

```
package ru.olegkravtsov.lab16

import androidx.navigation.fragment.findNavController

class FoxFragment : BaseStoryFragment() {
    override val imageRes = R.drawable.fox
    override val storyTextRes = R.string.fox_story

    override val choices = listOf(
        Choice(R.string.choice_obey) {

findNavController().navigate(R.id.action_foxFragment_to_finalBadFragment)
        },
        Choice(R.string.choice_run) {

findNavController().navigate(R.id.action_foxFragment_to_finalGoodFragment)
        }
    )
}
```

– FinalBadFragment - плохая концовка

```
package ru.olegkravtsov.lab16

import androidx.navigation.fragment.findNavController

class FinalBadFragment : BaseStoryFragment() {
    override val imageRes = R.drawable.final_bad
    override val storyTextRes = R.string.final_bad_story

    override val choices = listOf(
        Choice(R.string.choice_retry) {

findNavController().navigate(R.id.action_finalBadFragment_to_introFragment)
        }
    )
}
```

– FinalGoodFragment - хорошая концовка

```
package ru.olegkravtsov.lab16

import androidx.navigation.fragment.findNavController

class FinalGoodFragment : BaseStoryFragment() {
    override val imageRes = R.drawable.final_good
    override val storyTextRes = R.string.final_good_story

    override val choices = listOf(
        Choice(R.string.choice_restart) {

findNavController().navigate(R.id.action_finalGoodFragment_to_introFragment)
        }
    )
}
```

5. В файле nav_graph.xml определен граф навигации приложения с указанием всех фрагментов и действий перехода между ними:

```
<?xml version="1.0" encoding="utf-8"?>
<navigation 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:id="@+id/nav_graph"
    app:startDestination="@id/introFragment">

    <fragment
        android:id="@+id/introFragment"
        android:name="ru.olegkravtsov.lab16.IntroFragment"
        android:label="IntroFragment"
        tools:layout="@layout/fragment_story">
        <action
            android:id="@+id/action_introFragment_to_hareFragment"
            app:destination="@id/hareFragment" />
        <action
            android:id="@+id/action_introFragment_to_finalBadFragment"
            app:destination="@id/finalBadFragment" />
    </fragment>

    <fragment
        android:id="@+id/hareFragment"
        android:name="ru.olegkravtsov.lab16.HareFragment"
        android:label="HareFragment"
        tools:layout="@layout/fragment_story">
        <action
            android:id="@+id/action_hareFragment_to_wolfFragment"
            app:destination="@id/wolfFragment" />
        <action
            android:id="@+id/action_hareFragment_to_finalBadFragment"
            app:destination="@id/finalBadFragment" />
    </fragment>

    <fragment
        android:id="@+id/wolfFragment"
        android:name="ru.olegkravtsov.lab16.WolfFragment"
        android:label="WolfFragment"
        tools:layout="@layout/fragment_story">
        <action
            android:id="@+id/action_wolfFragment_to_bearFragment"
            app:destination="@id/bearFragment" />
        <action
            android:id="@+id/action_wolfFragment_to_finalBadFragment"
            app:destination="@id/finalBadFragment" />
    </fragment>

    <fragment
        android:id="@+id/bearFragment"
```

```
        android:name="ru.olegkravtsov.lab16.BearFragment"
        android:label="BearFragment"
        tools:layout="@layout/fragment_story">
        <action
            android:id="@+id/action_bearFragment_to_foxFragment"
            app:destination="@id/foxFragment" />
        <action
            android:id="@+id/action_bearFragment_to_finalBadFragment"
            app:destination="@id/finalBadFragment" />
    </fragment>

    <fragment
        android:id="@+id/foxFragment"
        android:name="ru.olegkravtsov.lab16.FoxFragment"
        android:label="FoxFragment"
        tools:layout="@layout/fragment_story">
        <action
            android:id="@+id/action_foxFragment_to_finalBadFragment"
            app:destination="@id/finalBadFragment" />
        <action
            android:id="@+id/action_foxFragment_to_finalGoodFragment"
            app:destination="@id/finalGoodFragment" />
    </fragment>

    <fragment
        android:id="@+id/finalBadFragment"
        android:name="ru.olegkravtsov.lab16.FinalBadFragment"
        android:label="FinalBadFragment"
        tools:layout="@layout/fragment_story">
        <action
            android:id="@+id/action_finalBadFragment_to_introFragment"
            app:destination="@id/introFragment" />
    </fragment>

    <fragment
        android:id="@+id/finalGoodFragment"
        android:name="ru.olegkravtsov.lab16.FinalGoodFragment"
        android:label="FinalGoodFragment"
        tools:layout="@layout/fragment_story">
        <action
            android:id="@+id/action_finalGoodFragment_to_introFragment"
            app:destination="@id/introFragment" />
    </fragment>

</navigation>
```

6. Реализована MainActivity, которая служит контейнером для навигационного хоста. Активность не требует дополнительной логики, так как вся навигация управляется через Navigation Component:

```
package ru.olegkravtsov.lab16

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

7. Разработан макет activity_main.xml для основной активности:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <androidx.fragment.app.FragmentContainerView
        android:id="@+id/nav_host_fragment"
        android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:defaultNavHost="true"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:navGraph="@navigation/nav_graph" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

8. Создан макет fragment_story.xml для отображения содержимого фрагментов:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="@dimen/default_padding">

    <androidx.core.widget.NestedScrollView
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:fillViewport="true">

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

            <ImageView
                android:id="@+id/storyImage"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:adjustViewBounds="true"
                android:maxHeight="@dimen/image_max_height"
                android:scaleType="fitCenter"
                android:contentDescription="@string/story_image" />

            <TextView
                android:id="@+id/storyText"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="@dimen/element_spacing"
                android:textSize="@dimen/text_size_medium"
                android:lineSpacingExtra="@dimen/line_spacing"
                android:textColor="@color/black" />

        </LinearLayout>

    </androidx.core.widget.NestedScrollView>

    <LinearLayout
        android:id="@+id/buttonsContainer"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_marginTop="@dimen/element_spacing" />

</LinearLayout>
```

Результат работы

Приложение успешно запускается и отображает начальный фрагмент с иллюстрацией и текстом первой сцены сказки "Колобок" (рис. 1). В нижней части экрана расположены кнопки выбора дальнейшего развития сюжета.

При нажатии на кнопки выбора происходит плавный переход между фрагментами (рис. 2, рис. 3, рис. 4). Каждый фрагмент содержит соответствующую иллюстрацию и текст части сказки. Навигация между фрагментами осуществляется с помощью компонента Navigation, что обеспечивает правильную работу жеста "Назад".

Рисунок 1 – Главный экран приложения с начальной сценой сказки

Рисунок 2 – Экран встречи с зайцем и варианты выбора действий

Рисунок 3 – Экран встречи с лисой и финальный выбор сюжета

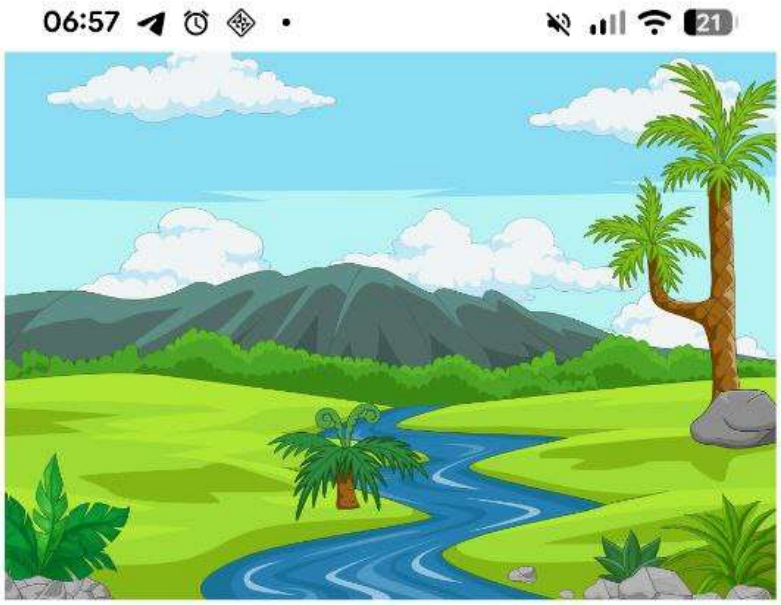


Рисунок 4 – Экран с хорошей концовкой сказки

Выводы

В ходе выполнения лабораторной работы было разработано приложение для интерактивного повествования сказки "Колобок" с использованием компонента Android Jetpack Navigation. Были освоены принципы навигации между фрагментами, создания навигационных графов и работы с NavController. Приложение демонстрирует правильную организацию навигации между экранами, сохранение состояния и обработку переходов. Полученные навыки позволяют создавать сложные сценарии навигации в Android-приложениях с соблюдением рекомендуемых архитектурных паттернов. Использование компонента Navigation значительно упрощает управление переходами между экранами и обеспечивает согласованное поведение приложения.