Лабароторная работа №3 выполнил Кучеренко Глеб

Чтобы из MainActivity запустить MainActivity2, надо вызвать метод startActivity():

Интент – это объект для обмена между активити, который абстрактно представляет собой намерение выполнить какое-либо действие. В основном интенты используются для запуска активити. Как только интент отправляется, его получает система Android, и считывает информацию в нем. Для открытия второго экрана, нам нужно создать и отправить объект Intent с указанием активити, которое нужно открыть. Затем вызвать метод startActivity() с передачей объекта Intent, который отправит это сообщение фреймворку Android, который откроет это активити.

Мы можем передать числовые значения между активностями так же, как и строковые, но с использованием другого метода. Вот пример модифицированного кода для MainActivity, где мы будем передавать целое число (например, возраст):

MainActivity:

```
import android.content.Intent
import android.os.Bundle
import android.widget.Button
import androidx.appcompat.app.AppCompatActivity
class MainActivity : AppCompatActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)

    val btn: Button = findViewById(R.id.btnIntent)
    btn.setOnClickListener {
      val intent = Intent(this, MainActivity2::class.java)
      intent.putExtra("text", "Измененный текст") // Ключ с названием "text"
      intent.putExtra("age", 25) // Передаем числовое значение "age"
      startActivity(intent)
    }
  }
```

```
}
```

MainActivity2:

```
import android.os.Bundle
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity
class MainActivity2 : AppCompatActivity() {
   override fun onCreate(savedInstanceState: Bundle?) {
      super.onCreate(savedInstanceState)
      setContentView(R.layout.activity_main2)
      val text: TextView = findViewById(R.id.tv)
      val message = intent.getStringExtra("text") // Получаем строку
      val age = intent.getIntExtra("age", 0) // Получаем целое число с ключом "age"
      text.text = "$message, твой возраст: $age" // Выводим полученные данные на
экран
   }
```

Задание 2: Использование ViewBinding

ViewBinding — это более современный и более безопасный способ работы с пользовательским интерфейсом в Android. Он помогает избежать ошибок, связанных с неправильными идентификаторами представлений и потенциалом возникновения NullPointerException, который вы можете столкнуться при использовании findViewById.

Для использования ViewBinding нужно сделать следующее:

В build.gradle (Module) добавьте следующую строку в раздел android:

```
viewBinding {
   enabled = true
}
```

В данном примере мы создаем Intent, используя putExtra для передачи как строкового, так и числового значения. В MainActivity2 мы получаем данные с помощью методов getStringExtra и getIntExtra. Обратите внимание на то, что getIntExtra принимает второй аргумент — значение по умолчанию на случай, если ключ не будет найден.

Задание 2: Использование ViewBinding

ViewBinding — это более современный и более безопасный способ работы с пользовательским интерфейсом в Android. Он помогает избежать ошибок, связанных с неправильными идентификаторами представлений и потенциалом возникновения NullPointerException, который вы можете столкнуться при использовании findViewById.

Для использования ViewBinding нужно сделать следующее:

- В build.gradle (Module) добавьте следующую строку в раздел android:

```
viewBinding {
    enabled = true
}
```

- Измените код активностей для использования ViewBinding. Вот как это будет выглядеть:

MainActivity с ViewBinding:

```
import android.content.Intent
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import com.example.yourapp.databinding.ActivityMainBinding // Импортируйте ваш сгенерированный класс ViewBinding


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


    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)


        binding.btnIntent.setOnClickListener {
```

```
        val intent = Intent(this, MainActivity2::class.java)
        intent.putExtra("text", "Измененный текст")
        intent.putExtra("age", 25)
        startActivity(intent)
    }
  }
}
```

MainActivity с ViewBinding2:

```
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import com.example.yourapp.databinding.ActivityMain2Binding // Импортируйте ваш сгенерированный класс ViewBinding

class MainActivity2 : AppCompatActivity() {
  private lateinit var binding: ActivityMain2Binding

  override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    binding = ActivityMain2Binding.inflate(layoutInflater)
    setContentView(binding.root)

    val message = intent.getStringExtra("text")
    val age = intent.getIntExtra("age", 0)

    binding.tv.text = "$message, твой возраст: $age"
  }
```

}

Разница между findViewById и ViewBinding

Типобезопасность: ViewBinding предоставляет типизированные ссылки на представления. Это означает, что компилятор проверяет идентификаторы представлений во время компиляции, а не во время выполнения, как в случае с findViewById.

Производительность: ViewBinding более эффективен, т.к. минимизирует затраты на поиск представлений.

Упрощение кода: Использование ViewBinding позволяет избежать лишних вызовов findViewById, что делает код более читаемым и лаконичным.

Таким образом, ViewBinding становится более предпочтительным выбором для работы с пользовательскими интерфейсами в Android, так как улучшает безопасность, производительность и читабельность кода.