[https://youtu.be/EkXD07ps6Qo](https://youtu.be/EkXD07ps6Qo" \t "_blank)

**Передача данных между экранами — пунктами назначения. Android Navigation. Bundle vs Safe Args**

**На этом уроке**

На этом уроке рассмотрим возможности передачи данных между экранами – пунктами назначения навигации в андроид-приложении. Создадим приложение с двумя экранами. На первом экране будет поле для ввода имени и кнопка отправки, а на втором экране будет отображаться приветствие с именем, которое мы указали. Таким образом, мы передадим данные (имя) с первого экрана на второй.

**Передача данных между экранами**

В процессе работы приложения часто возникает необходимость передавать данные между экранами.  Мы рассматривали возможность передачи данных между активити с помощью интента. Для передачи данных между фрагментами такой способ не подходит, поскольку часто фрагменты отображаются в одном и том же активити.

**Bundle или Safe Args?**

В этом уроке мы рассмотрим два способа передачи данных между фрагментами: традиционный – с помощью наборов данных Bundle и типобезопасный – при помощи безопасных аргументов SafeArgs. Первый способ относительно прост – создаем набор данных «ключ-значение» типа Bundle и передаем через action в первом фрагменте, и извлекаем во втором фрагменте.  
Второй способ потребует немного больше кода. На первый взгляд он может показаться сложнее, поскольку используется кодогенерация – среда разработки создает необходимые классы вместо вас. Но мы попробуем разобраться и вы увидите, что ничего особо сложного там нет. По сути, SafeArgs – просто обертка над Bundle. Тем не менее, разработчики настоятельно рекомендуют применять именно SafeArgs, как типобезопасный способ передачи данных между фрагментами в процессе навигации.

**Создаем проект**

Откройте среду разработки Android Studio и создайте новый проект с использованием шаблона Empty views Activity.

**Создаем граф навигации**

Далее перейдите в папку **res** и создайте в ней папку **navigation**.  Внутри папки **navigation** создайте Navigation Resource File с именем nav\_graph.xml и корневым элементом <navigation>.

Если вы забыли добавить в проект необходимые для поддержки навигации библиотеки – Android Studio предложит это сделать за вас, показав предупреждение.

**Добавляем пункты назначения – фрагменты**

Добавьте новые пункты назначения. Для этого:

1. В окне редактора дизайна нажмите кнопку «New destination»
2. Выберите «Create new destination»
3. Далее в окне добавления фрагмента выберите Fragment (Blank):

Создайте таким образом два фрагмента:

FragmentOne

FragmentTwo

Чтобы фрагменты из графа навигации отображались на экране, не забудьте добавить в макет главного активити activity\_main.xml компонент fragment — хост навигации вместо TextView:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | <?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:layout\_width="match\_parent"      android:layout\_height="match\_parent"      tools:context=".MainActivity">        <fragment          android:id="@+id/nav\_graph"          android:name="androidx.navigation.fragment.NavHostFragment"          android:layout\_width="match\_parent"          android:layout\_height="match\_parent"          app:defaultNavHost="true"          app:navGraph="@navigation/nav\_graph" />    </androidx.constraintlayout.widget.ConstraintLayout> |

**Создаем action**

Вернемся в nav\_graph.xml и добавим action для перехода. Можно просто соединить фрагменты стрелкой  на экране редактора дизайна, и в коде графа навигации добавится секция <action>:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | <?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/fragmentOne">        <fragment          android:id="@+id/fragmentOne"          android:name="info.fandroid.myapplication15.FragmentOne"          android:label="fragment\_one"          tools:layout="@layout/fragment\_one" >          <action              android:id="@+id/action\_fragmentOne\_to\_fragmentTwo"              app:destination="@id/fragmentTwo" />      </fragment>      <fragment          android:id="@+id/fragmentTwo"          android:name="info.fandroid.myapplication15.FragmentTwo"          android:label="fragment\_two"          tools:layout="@layout/fragment\_two" >      </fragment>  </navigation> |

**Добавим аргумент**

Для передачи данных нужно использовать элемент <argument>, добавим его в граф навигации. Для этого выделите второй фрагмент в редакторе дизайна и справа в панели атрибутов на вкладке Arguments нажмите плюс. Укажите имя аргумента, например, MyArg и значение по умолчанию. Тип можно не указывать. Я передам в качестве дефолтного значения текст «Hello, Android!». Значение по умолчанию будет отображаться, если мы ничего не передадим из первого фрагмента во второй.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | <fragment      android:id="@+id/fragmentTwo"      android:name="info.fandroid.myapplication15.FragmentTwo"      android:label="fragment\_two"      tools:layout="@layout/fragment\_two" >      <argument          android:name="MyArg"          android:defaultValue="Hello, Android!" />  </fragment> |

**Подготовим макеты фрагментов**

Измените макет разметки первого фрагмента fragment\_one.xml таким образом:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44 | <?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:layout\_width="match\_parent"      android:layout\_height="match\_parent"      tools:context=".FragmentOne">        <TextView          android:id="@+id/tvFragmentOne"          android:textSize="48sp"          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:text="@string/text\_what\_your\_name"          app:layout\_constraintBottom\_toTopOf="@+id/editText"          app:layout\_constraintEnd\_toEndOf="parent"          app:layout\_constraintStart\_toStartOf="parent"          app:layout\_constraintTop\_toTopOf="parent" />        <EditText          android:id="@+id/editText"          android:layout\_width="0dp"          android:layout\_height="wrap\_content"          app:layout\_constraintBottom\_toBottomOf="parent"          app:layout\_constraintEnd\_toEndOf="parent"          app:layout\_constraintHorizontal\_bias="0.0"          app:layout\_constraintStart\_toStartOf="parent"          app:layout\_constraintTop\_toTopOf="parent"          app:layout\_constraintVertical\_bias="0.53"          app:layout\_constraintEnd\_toStartOf="@+id/imgButton"/>        <ImageButton          android:id="@+id/imgButton"          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:src="@drawable/baseline\_send\_black\_36"          app:layout\_constraintBottom\_toBottomOf="@+id/editText"          app:layout\_constraintEnd\_toEndOf="parent"          app:layout\_constraintHorizontal\_bias="0.0"          app:layout\_constraintStart\_toEndOf="@+id/editText"          app:layout\_constraintTop\_toTopOf="@+id/editText" />    </androidx.constraintlayout.widget.ConstraintLayout> |

Здесь мы оставили текстовое поле, которое теперь отображает текст «Ваше имя?». Ниже TextView добавлен EditText для возможности вписать туда имя, и кнопка отправки ImageButton.

Введенный на первом экране текст будем передавать во второй фрагмент для отображения в текстовом поле. Откройте макет второго фрагмента fragment\_two.xml и сделайте так, чтобы элемент TextView был посредине. Для этого можно заменить корневой FrameLayout, например, на ConstraintLayout. Также добавьте идентификатор для TextView.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | <?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:layout\_width="match\_parent"      android:layout\_height="match\_parent"      tools:context=".FragmentTwo">          <TextView          android:id="@+id/tvFragmentTwo"          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          app:layout\_constraintBottom\_toBottomOf="parent"          app:layout\_constraintEnd\_toEndOf="parent"          app:layout\_constraintStart\_toStartOf="parent"          app:layout\_constraintTop\_toTopOf="parent"          tools:text="@string/hello\_blank\_fragment"          android:textSize="48sp"/>    </androidx.constraintlayout.widget.ConstraintLayout> |

**Передаем данные через Bundle**

В классе FragmentOne переопределим метод onViewCreated и напишем в нем такой код:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | override fun onViewCreated(view: View, savedInstanceState: Bundle?) {     super.onViewCreated(view, savedInstanceState)        val imgButton : ImageButton = view.findViewById(R.id.imgButton)      val editText : EditText = view.findViewById(R.id.editText)      val bundle = Bundle()        imgButton.setOnClickListener {          val name = editText.text          val hello = "Привет, $name"          bundle.putString("MyArg", hello)          findNavController().navigate(R.id.fragmentTwo, bundle)      }   } |

Обратите внимание – переменные кнопки и поля для ввода текста мы инициализируем через метод findViewById, как мы это делали на первых уроках. Этот способ требует больше повторяющегося кода в отличие от использования ViewBinding и DataBinding для привязки элементов разметки, однако многие программисты пользуются именно этим способом, как самым простым и надежным.

Объявим также переменную для набора данных Bundle.

Далее присваиваем слушатель кнопке, и по ее нажатию производим следующие действия:

* Инициализируем переменную name и сохраняем в нее текст из поля ввода;
* Переменная hello уже содержит текст и добавляет в него текст переменной name;
* В переменную bundle передаем переменную hello с ключем «MyArg»
* Обращаемся к контроллеру навигации и реализуем переход к FragmentTwo с передачей идентификатора макета фрагмента и bundle в метод navigate контроллера.

**Принимаем данные через Bundle**

В коде класса FragmentTwo также переопределяем метод onViewCreated и пишем в него код:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | override fun onViewCreated(view: View, savedInstanceState: Bundle?) {      super.onViewCreated(view, savedInstanceState)        val tv :TextView = view.findViewById(R.id.tvFragmentTwo)      val text = arguments?.getString("MyArg")      tv.text = text    } |

Здесь мы инициализируем TextView и переменную text. В эту переменную мы сохраняем значение из переданного Bundle, обращаясь к нему через его геттер arguments, и получаем строку с ключом «MyArg». Далее передаем ее для отображения в текстовое поле.

**Тестирование приложения**

Запустите приложения на эмуляторе или смартфоне. На первом экране с полем ввода и кнопкой введите имя и нажмите кнопку отправки. Откроется второй экран, где будет отображен текст приветствия с введенным именем.

**Подключение в проект Safe Args**

Несмотря на простоту этого способа, официальные разработчики все же рекомендуют выполнять передачу данных между фрагментами посредством Safe Args, так как этот способ является типобезопасным.

Чтобы добавить поддержку Safe Args в проект, в файле build.gradle верхнего уровня пропишите:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | buildscript {      repositories {          google()      }      dependencies {          def nav\_version = "2.3.3"   classpath "androidx.navigation:navigation-safe-args-gradle-plugin:$nav\_version"      }  } |

 Вы также должны применить один из двух доступных плагинов.

Если ваш проект на языке Kotlin, добавьте эту строку в файл build.gradle модуля app вашего приложения в раздел plugin:



|  |  |
| --- | --- |
| 1 | **id 'androidx.navigation.safeargs'** |

Также вы должны для поддержки AndroidX указать в файле gradle.properties .

android.useAndroidX=true

**Дальше описывается альтернатива, поэтому можно не делать!!!!!!**

**Передача данных через Safe Args**

Теперь изменим код фрагментов.

Метод onViewCreated класса FragmentOne:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | override fun onViewCreated(view: View, savedInstanceState: Bundle?) {     super.onViewCreated(view, savedInstanceState)        val imgButton : ImageButton = view.findViewById(R.id.imgButton)      val editText : EditText = view.findViewById(R.id.editText)      //val bundle = Bundle()        imgButton.setOnClickListener {          val name = editText.text          val hello = "Привет, $name"           bundle.putString("MyArg", hello)          findNavController().navigate(R.id.fragmentTwo, bundle)        }    } |

Поскольку мы теперь не работаем с Bundle напрямую, в методе onViewCreated закомментируем строку объявления его переменной.

Далее в слушателе нажатия кнопки вместо сохранения аргумента в bundle и передачи его контроллеру, будем использовать сгенерированный средой разработки класс FragmentOneDirections. В его функцию actionFragmentOneToFragmentTwo передаем строковую переменную hello и затем это все передаем методу navigate контроллера навигации.

А если мы откроем класс FragmentOneDirections, то увидим, что внутри он использует все тот же Bundle.

Теперь изменим код метода onViewCreated класса FragmentTwo:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | override fun onViewCreated(view: View, savedInstanceState: Bundle?) {      super.onViewCreated(view, savedInstanceState)        val tv :TextView = view.findViewById(R.id.tvFragmentTwo)      //val text = arguments?.getString("MyArg")      val args: FragmentTwoArgs by navArgs()      val text = args.MyArg      tv.text = text    } |

 Вместо получения строки из Bundle инициализируем переменную класса FragmentTwoArgs, который хранит данные, переданные нами из первого фрагмента. Мы можем извлечь данные, обращаясь к свойству экземпляра класса по имени аргумента, которое мы указали при его создании в графе навигации. Здесь мы передаем его в текстовое поле.

Если открыть FragmentTwoArgs, мы увидим, что это дата-класс с публичной переменной, имя которой совпадает с именем нашего аргумента. Эта переменная получает свое значение посредством Bundle, а также имеет значение по умолчанию – это фраза «Hello, Android!» —  та самая, которую мы указали как значение по умолчанию аргумента в графе навигации.

Оба класса – FragmentTwoArgs и FragmentOneDirections – сгенерированы средой разработки, их можно найти в папке build модуля app нашего проекта.

**Тестирование приложения**

При запуске приложения открывается первый экран с предложением ввести имя. После ввода и нажимаем кнопку отправки и видим второй экран с приветствием, в котором использовано введенное имя. Таким образом, данные с первого экрана передаются во второй.

Если в коде второго фрагмента ничего не передать в переменную action, то на втором экране будет отображаться значение аргумента по умолчанию.