7. Навигация в приложениях

Взаимодействие между Activity и Fragment.

Введение в Jetpack Navigation Component.

Цели лекции:

1. Познакомиться с основами навигации в Android-приложениях.

2. Изучить взаимодействие между `Activity` и `Fragment`.

3. Изучить Jetpack Navigation Component для упрощения управления навигацией.

В этой лекции мы рассмотрим навигацию в Android-приложениях, включая взаимодействие между `Activity` и `Fragment`. Мы узнаем, как организовать переходы между экранами и передавать данные между компонентами. Также будет представлено введение в Jetpack Navigation Component, который упрощает управление навигацией, улучшая пользовательский опыт и упрощая разработку. Лекция позволит вам понять, как эффективно использовать навигационные компоненты для создания удобных и интуитивно понятных интерфейсов в приложениях.

Навигация в приложениях

Навигация является ключевым аспектом пользовательского опыта в мобильных приложениях. В этой лекции мы изучим, как организовать переходы между различными экранами, а также как использовать взаимодействие между `Activity` и `Fragment`. Мы также познакомимся с Jetpack Navigation Component, который упрощает управление навигацией в Android-приложениях.

1. Основы навигации в Android

Навигация в Android-приложениях часто осуществляется с помощью `Activity` и `Fragment`.

- Activity представляет собой отдельный экран в приложении, и каждое действие (например, показ списка, редактирование данных) может обрабатываться отдельной активностью.

- Fragment — это компонент пользовательского интерфейса, который можно вставить в `Activity`. Он позволяет создавать многоэкранные интерфейсы и повторно использовать интерфейсные элементы.

Переход между `Activity` осуществляется с помощью `Intents`, которые позволяют запускать новую активность, передавая данные при необходимости. Например:

```java

Intent intent = new Intent(CurrentActivity.this, NewActivity.class);

intent.putExtra("key", "value");

startActivity(intent);

```

При использовании `Fragment` навигация осуществляется через `FragmentManager`, который управляет добавлением, заменой и удалением фрагментов.

2. Взаимодействие между Activity и Fragment

Взаимодействие между `Activity` и `Fragment` осуществляется через интерфейсы и методы обратного вызова. Например, для передачи данных из `Fragment` в `Activity`, можно создать интерфейс в `Fragment` и реализовать его в `Activity`. Это позволит `Fragment` сообщать `Activity` о каких-либо действиях или передавать данные.

```java

public interface OnDataPass {

void onDataPass(String data);

}

```

В `Fragment` мы вызываем метод:

```java

dataPasser.onDataPass(data);

```

А в `Activity` мы реализуем этот метод:

```java

@Override

public void onDataPass(String data) {

// Обработка данных

}

```

3. Введение в Jetpack Navigation Component

Jetpack Navigation Component — это мощный инструмент для управления навигацией в Android-приложениях. Он позволяет создать граф навигации, который определяет все маршруты приложения, облегчая управление переходами и передачу данных.

3.1 Преимущества Navigation Component

- Упрощение навигации: Вы можете визуально организовать навигацию с помощью `NavGraph`, где задаете все возможные переходы между экранами.

- Управление аргументами: Навигационный компонент упрощает передачу данных между `Fragment` и `Activity`, позволяя определять аргументы в графе навигации.

- Интеграция с `ViewModel`: Navigation Component работает в паре с `ViewModel`, что упрощает управление состоянием приложения.

3.2 Использование NavController

`NavController` управляет навигацией и выполняет переходы между экранами. Чтобы настроить его, нужно добавить `NavHostFragment` в ваш XML-файл макета:

```xml

<androidx.navigation.fragment.NavHostFragment

android:id="@+id/nav\_host\_fragment"

android:name="androidx.navigation.fragment.NavHostFragment"

app:navGraph="@navigation/nav\_graph"

app:defaultNavHost="true"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" />

```

После этого вы можете использовать `NavController` для выполнения переходов:

```java

NavController navController = Navigation.findNavController(this, R.id.nav\_host\_fragment);

navController.navigate(R.id.action\_firstFragment\_to\_secondFragment);

```

Заключение

В этой лекции мы изучили основы навигации в Android-приложениях, включая взаимодействие между `Activity` и `Fragment`. Мы также познакомились с Jetpack Navigation Component, который упрощает управление навигацией и позволяет создавать более удобные и интуитивно понятные интерфейсы. Понимание этих концепций поможет вам создавать более эффективные и привлекательные приложения для пользователей.

<https://youtu.be/tyxt87DX7mw?feature=shared>

<https://youtu.be/ntZFCNCCTeI?feature=shared>

<https://youtu.be/KYTHP9WtTHY?feature=shared>