Android Navigation Graphs

A navigation graph is an XML resource file (usually named nav_graph.xml) that defines the navigation structure of your app. It typically includes destinations (screens or fragments) and the actions that define how the user can navigate between these destinations

➢ Destinations:

In the context of Android Navigation, a "destination" refers to a specific screen or UI element that the user can navigate to. Destinations are represented by fragments, activities, or nested navigation graphs within the navigation graph XML file.

1. Fragment Destination:

- **Represents a UI screen or a portion of the app's user interface.**
- **Example:**

xml

```
<fragment
   android:id="@+id/firstFragment"
   android:name="com.example.FirstFragment"
   android:label="First Fragment">
</fragment>
```

2. Activity Destination:

- **Represents a screen or component implemented as a separate activity.**
- **Example:**

xml

```
<activity
   android:id="@+id/secondActivity"
   android:name="com.example.SecondActivity"
   android:label="Second Activity">
</activity>
```

3. Nested Graph Destination:

- Represents a separate navigation graph that is nested within the current navigation graph.
- Example:

xml

```
<navigation
   android:id="@+id/nestedGraph"
   android:label="Nested Graph">
   <! Define nested destinations and actions here >
</navigation>
```

Actions:

Actions define the possible paths or transitions between destinations in the navigation graph. They specify how the user can navigate from one destination to another.

1. Navigate Action:

- Represents a direct navigation from one destination to another.
- Example:

xml

```
<action
   android:id="@+id/actionFirstToSecond"
   app:destination="@id/secondFragment"
   app:enterAnim="@anim/slide_in"
   app:exitAnim="@anim/slide_out" />
```

In code:

java

```
NavController navController = Navigation.findNavController(this, R.id.nav_host_fragment);
navController.navigate(R.id.actionFirstToSecond);
```

2. Deep Link Action:

- Represents a navigation triggered by a URI or a deep link.
- Example:

```
xml
<deepLink
   app:uri="example.com/second"
   app:action="@action/actionFirstToSecond" />
```

3. Conditional Action:

- Represents a conditional navigation based on certain conditions.
- Example:

```
xml
<action
   android:id="@+id/actionConditional"
   app:destination="@id/conditionalDest"
   app:popUpTo="@+id/conditionalDest"
   app:popUpToInclusive="true" />
```

This action pops up the back stack up to and including the specified destination if certain conditions are met.

How They Work Together:

In a navigation graph, destinations are connected by actions, forming a visual representation of the app's navigation flow. When a user triggers a navigation event, the associated action determines how the app transitions from the current destination to the next.

For example, if you have a button in `FirstFragment` and you want to navigate to `SecondFragment` when the button is clicked, you would define a `Navigate Action` in the navigation graph linking these two destinations. When the button is clicked, you call `navController.navigate(R.id.actionFirstToSecond)` to initiate the navigation.

This separation of destinations and actions in a navigation graph provides a clean and declarative way to define and visualize the app's navigation structure, making it easier to manage and maintain the navigation flow in a complex Android app.