Intents & Intro to Fragments

Multiple screens in an app

Sometimes app functionality may be separated into multiple screens.

Examples:

- View details of a single item (for example, product in a shopping app)
- Create a new item (for example, new email)
- Show settings for an app
- Access services in other apps (for example, photo gallery or browse documents)

Intent

An **Intent** is a messaging object you can use to request an action from another app component

An `Intent` usually has two primary pieces of information:

- ○ Action to be performed (for example, `ACTION_VIEW`, `ACTION_EDIT`, `ACTION_MAIN`)
- ○ Data to operate on (for example, a person's record in the contacts database)

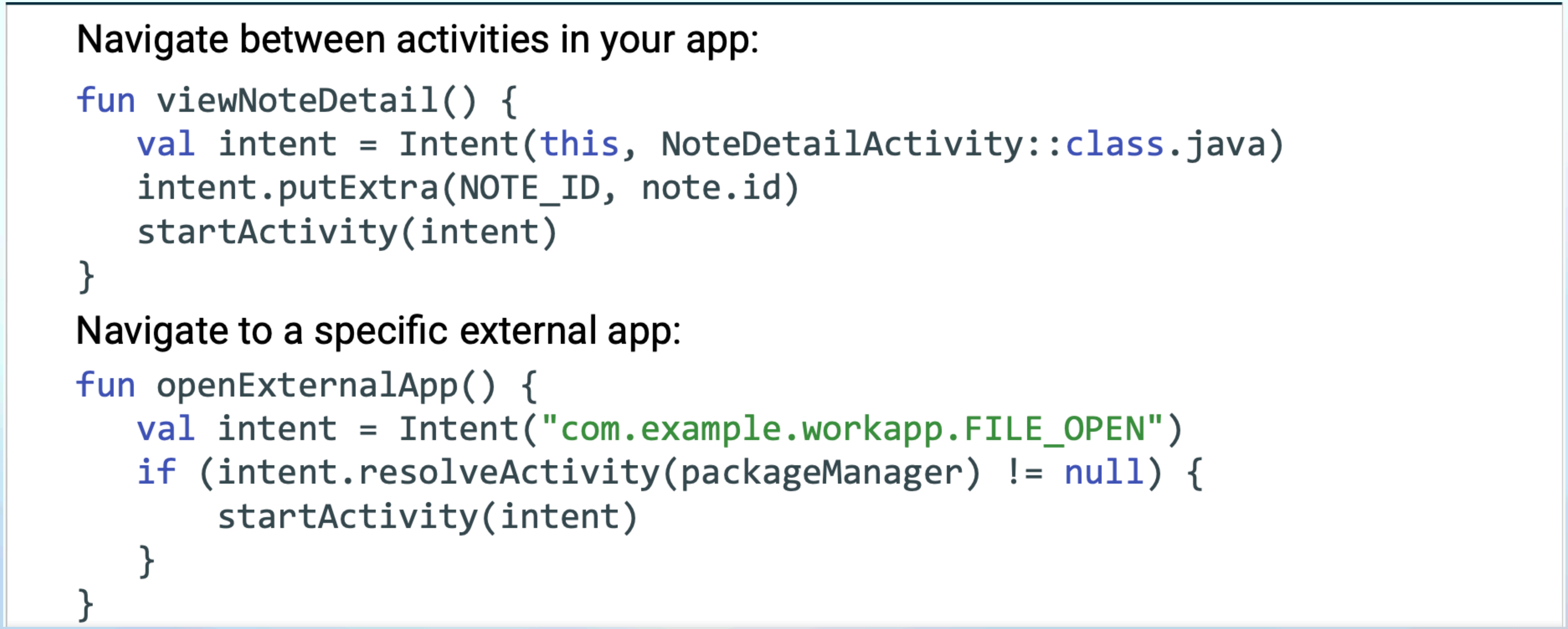
- Commonly used to specify a request to transition to another Activity

For more information https://developer.android.com/guide/components/intents-filters

Explicit intent

- Fulfills a request **using a specific component**
- Navigates internally to an Activity in your app
- Navigates to a specific third-party app or another app you've written

Explicit intent examples



Navigate between activities in your app:

```
fun viewNoteDetail() {
   val intent = Intent(this, NoteDetailActivity::class.java)
   intent.putExtra(NOTE_ID, note.id)
   startActivity(intent)
}
```

Navigate to a specific external app:

```
fun openExternalApp() {
   val intent = Intent("com.example.workapp.FILE_OPEN")
   if (intent.resolveActivity(packageManager) != null) {
       startActivity(intent)
   }
}
```

Implicit intent

- Provides generic action the app can perform
- Resolved using mapping of the data type and action to known components
- Allows any app that matches the criteria to handle the request

Implicit intent example

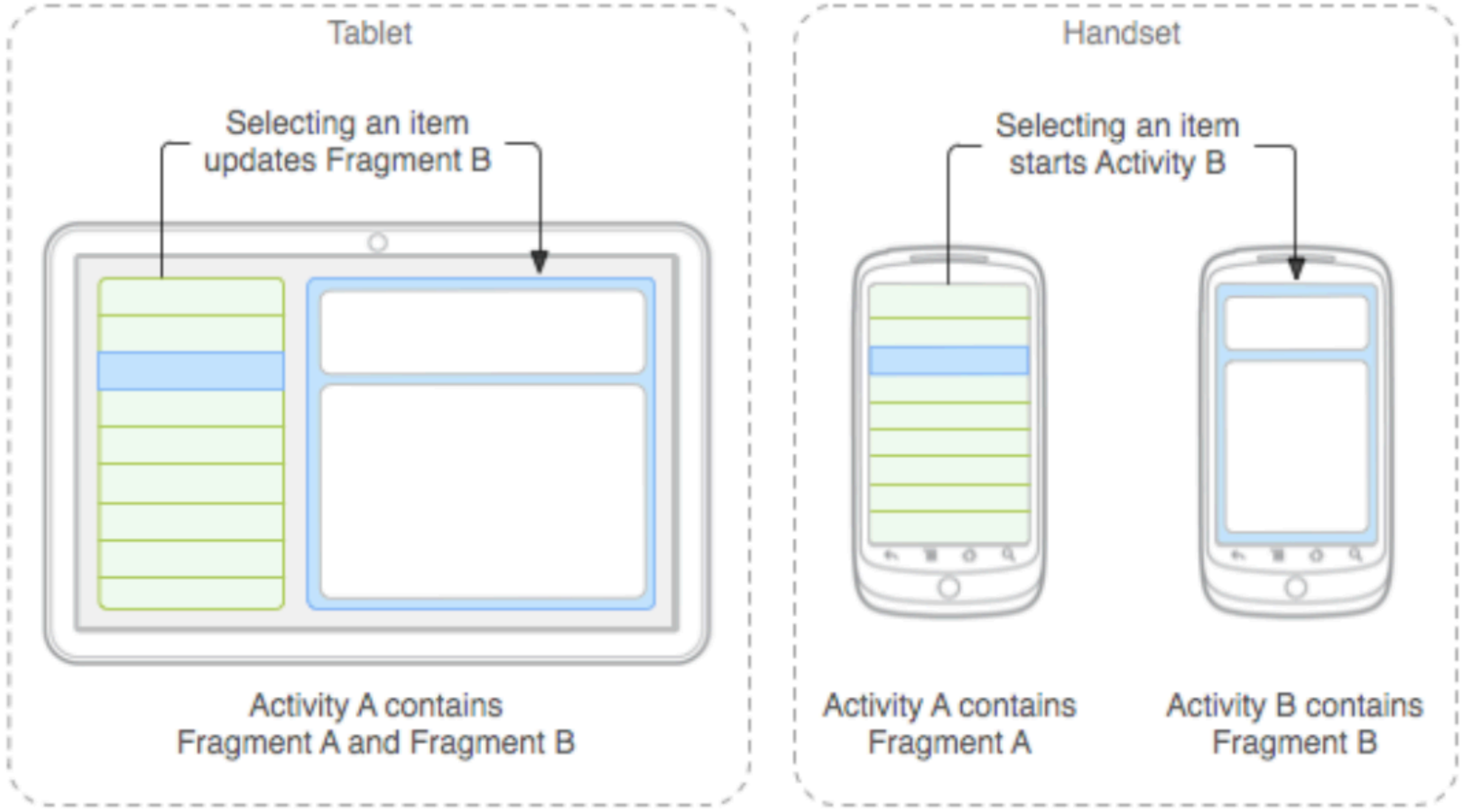
```kotlin
fun sendEmail() {
    val intent = Intent(Intent.ACTION_SEND)
    intent.type = "text/plain"
    intent.putExtra(Intent.EXTRA_EMAIL, emailAddresses)
    intent.putExtra(Intent.EXTRA_TEXT, "How are you?")

    if (intent.resolveActivity(packageManager) != null) {
        startActivity(intent)
    }
}
```

Fragments

Fragment

- Represents a behavior or portion of the UI in an activity ("microactivity")
- Must be hosted in an activity
- Lifecycle tied to host activity's lifecycle
- Have their own layout and behavior
- Can be added, removed, or replaced dynamically
- Must always be hosted in an Activity
- Can communicate with the host Activity and other Fragments

Fragments for tablet layouts


Tablet
Selecting an item updates Fragment B
Activity A contains Fragment A and Fragment B
Handset
Selecting an item starts Activity B
Activity A contains Fragment A
Activity B contains Fragment B

Note about fragments

Use the AndroidX version of the `Fragment` class. (`androidx.fragment.app.Fragment`).

Don't use the platform version of the `Fragment` class (`android.app.Fragment`), which was deprecated.

Fragment vs Activity Lifecycle

- Fragment lifecycle is directly influenced by its host Activity
- Fragment lifecycle methods are called after the corresponding Activity methods
- Fragments can be added to the back stack for navigation history

Creating a basic fragment

Step 1: Create Fragment Layout (fragment_example.xml

```xml
<?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="16dp"
    android:background="#f0f0f0">

    <TextView
        android:id="@+id/fragment_header"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="This is a Fragment"
        android:textSize="24sp"
        android:textStyle="bold"
        android:layout_marginBottom="16dp" />

    <Button
        android:id="@+id/fragment_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Click Me" />

</LinearLayout>
```

Create a Fragment class

```
class ExampleFragment : Fragment() {
    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_example, container, false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        view.findViewById<Button>(R.id.fragment_button).setOnClickListener {
            Toast.makeText(context, "Button clicked in fragment", Toast.LENGTH_SHORT).show()
        }
    }
}
```

Adding Fragments to an Activity

```
<?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"

    android:layout_width="match_parent"

    android:layout_height="match_parent">

    <androidx.fragment.app.FragmentContainerView

        android:id="@+id/fragment_container"

        android:name="com.example.myapp.ExampleFragment"

        android:layout_width="match_parent"

        android:layout_height="300dp"

        app:layout_constraintTop_toTopOf="parent" />

———— Rest of the code————
```

Dynamic Addition

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        // Check if the activity is being created for the first time
        if (savedInstanceState == null) {
            val fragment = ExampleFragment()

            // Get the FragmentManager and start a transaction
            supportFragmentManager.beginTransaction()
                .add(R.id.fragment_container, fragment) // R.id.fragment_container is a FrameLayout
                .commit()
        }
        findViewById<Button>(R.id.add_fragment_button).setOnClickListener {
            val newFragment = ExampleFragment()
            supportFragmentManager.beginTransaction()
                .replace(R.id.fragment_container, newFragment)
                .addToBackStack(null) // Allows back button to pop the fragment
                .commit()
        }
    }
}
```

Thank you