

**Universidad Nacional Politécnica, UNP**

Programación Móvil

Investigación I: Fragmentos

Fecha: 22 de junio de 2024

Profesor: Luis Guido Calderon

Ingeniería en computación V año

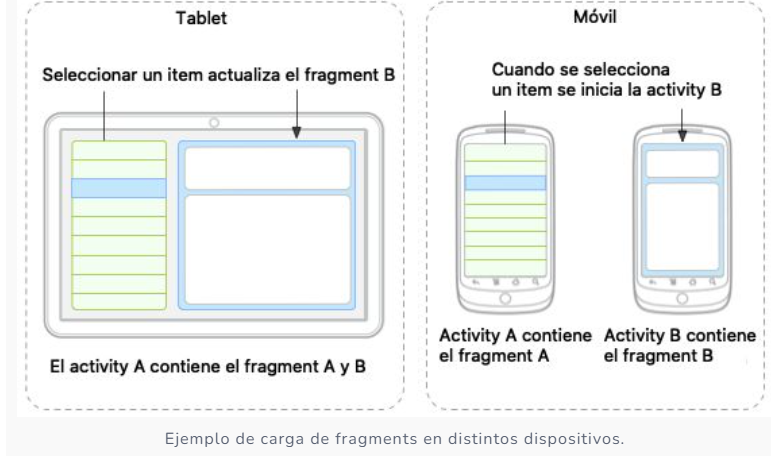
Joselin Raquel Morales Sotelo

1. **Fragmentos**

Un fragment representa una parte de reutilizable de la IU de tu app. Define y administra su propio diseño, tiene su propio ciclo de vida y puede administrar sus propios eventos de entradas. Los fragmentos no pueden existir solo, deben estar alojados por una actividad u otro fragmento.

Los fragmentos introducen el modularidad y la capacidad de reutilización en la IU de tu actividad, ya que te permiten dividir la IU en fragmentos separados. Las actividades son un lugar ideal para colocar elementos globales en la interfaz de usuario de tu app, como un panel lateral de navegación. En cambio, los fragmentos son más adecuados para definir y administrar la IU de una sola pantalla o de una parte de ella.

Tomemos como ejemplo una app que responda a varios tamaños de pantalla. En pantallas más grandes, te recomendamos que la app muestre un panel lateral de navegación estático y una lista en un diseño de cuadrícula. En pantallas más pequeñas, te recomendamos que la app muestre una barra de navegación inferior y una lista en un diseño lineal.

Administrar estas variaciones en la actividad es difícil de controlar. Separar los elementos de navegación del contenido puede hacer que ese proceso sea más fácil de manejar. La actividad se encarga de mostrar la IU de navegación correcta, mientras que el fragmento muestra la lista con el diseño adecuado.

**Como crear un fragmento**

Un fragmento representa una parte modular de la interfaz de usuario dentro de una actividad. Un fragmento tiene su propio ciclo de vida, recibe sus propios eventos de entrada, y tú puedes agregar o quitar fragmentos mientras se ejecuta la actividad que lo contiene.

Los fragmentos requieren una dependencia en la biblioteca de fragmentos de AndroidX. Para incluir esta dependencia, debes agregar el repositorio de Maven de Google al archivo settings.gradle de tu proyecto.

Para incluir la biblioteca de AndroidX Fragment a tu proyecto, agrega las siguientes dependencias al archivo build.gradle de tu app:

Si deseas crear un fragmento, extiende la clase Fragment de AndroidX y anula sus métodos para insertar la lógica de tu app, de manera similar a cómo crearías una clase Activity. Para crear un fragmento mínimo que defina su propio diseño, proporciona el recurso de diseño de tu fragmento en el constructor básico, como se muestra en el siguiente ejemplo:



**Cómo crear una clase de fragmento**

Si deseas crear un fragmento, extiende la clase Fragment de AndroidX y anula sus métodos para insertar la lógica de tu app, de manera similar a cómo crearías una clase Activity. Para crear un fragmento mínimo que defina su propio diseño, proporciona el recurso de diseño de tu fragmento en el constructor básico, como se muestra en el siguiente ejemplo:

class ExampleFragment : Fragment(R.layout.example\_fragment)

La biblioteca de fragmentos también proporciona clases básicas de fragmentos más especializadas:

DialogFragment

Muestra un diálogo flotante. Usar esta clase a fin de crear un diálogo es una buena alternativa al uso de los métodos auxiliares de diálogo en la clase Activity, ya que los fragmentos administran automáticamente la creación y la limpieza del Dialog. Consulta Cómo mostrar diálogos con DialogFragment para obtener más información.

PreferenceFragmentCompat

Muestra una jerarquía de objetos Preference en forma de lista. Puedes usar PreferenceFragmentCompat a fin de crear una pantalla de configuración para tu app.

Cómo agregar un fragmento a una actividad

En general, tu fragmento debe estar incorporado dentro de una FragmentActivity de AndroidX para contribuir con una parte de la IU al diseño de esa actividad. FragmentActivity es la clase básica de AppCompatActivity, de modo que si ya creas una subclase de AppCompatActivity a fin de proporcionar retrocompatibilidad en tu app, no es necesario que cambies la clase básica de tu actividad.

Puedes agregar tu fragmento a la jerarquía de vistas de la actividad; para ello, define el fragmento en el archivo de diseño de tu actividad, o bien define un contenedor de fragmentos en el archivo de diseño de la actividad y, luego, agrega el fragmento de forma programática desde tu actividad. En cualquier caso, debes agregar una FragmentContainerView que defina la ubicación donde se debería colocar el fragmento dentro de la jerarquía de vistas de la actividad. Te recomendamos que siempre uses una FragmentContainerView como contenedor de fragmentos, ya que FragmentContainerView incluye correcciones específicas para fragmentos que no proporcionan otros grupos de vistas, como FrameLayout.

Cómo agregar un fragmento a través de XML

Para agregar un fragmento de forma declarativa al XML de diseño de tu actividad, usa un elemento FragmentContainerView.

A continuación, puedes ver un ejemplo de diseño de actividad que contiene un único FragmentContainerView:

<!-- res/layout/example\_activity.xml -->

<androidx.fragment.app.FragmentContainerView

xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@+id/fragment\_container\_view"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:name="com.example.ExampleFragment" />

El atributo android:name especifica el nombre de clase del Fragment cuya instancia se creará. Cuando el diseño de la actividad aumenta, se crea una instancia del fragmento especificado, se llama a onInflate () en el fragmento cuya instancia se acaba de crear y se crea una FragmentTransaction para agregar el fragmento al FragmentManager.

**Cómo guardar un estado con fragmentos**

bookmark\_border

Varias operaciones del sistema Android pueden afectar el estado de tu fragmento. Para garantizar que se guarde el estado del usuario, el framework de Android guarda y restablece automáticamente los fragmentos y la pila de actividades. Por lo tanto, debes asegurarte de guardar y restablecer también todos los datos de tu fragmento.

En la siguiente tabla, se describen las operaciones que hacen que el fragmento pierda su estado, junto con la posibilidad de que los diferentes tipos de estado se conserven a través de esos cambios. Los tipos de estado que se mencionan en la tabla son los siguientes:

Variables: Variables locales en el fragmento

View State: Cualquier dato que pertenezca a una o más vistas del fragmento

SavedState: Datos inherentes a esta instancia del fragmento que se deben guardar en onSaveInstanceState()

NonConfig: Datos extraídos de una fuente externa, como un servidor o repositorio local, o datos creados por el usuario que se envían al servidor una vez confirmados

A menudo, las Variables se tratan de la misma manera que el SavedState, pero la tabla siguiente distingue entre los dos para mostrar el efecto de las distintas operaciones en cada uno.