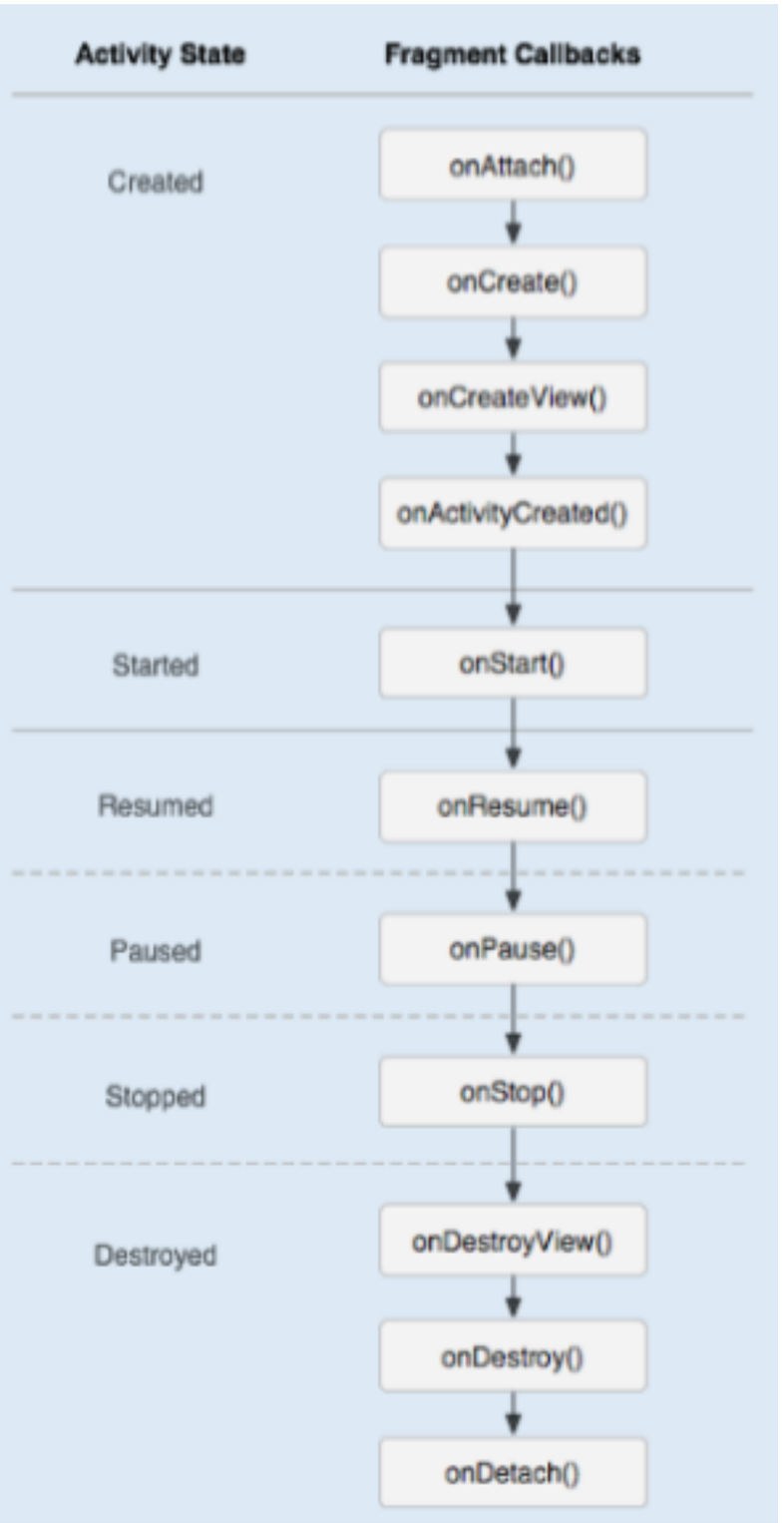
Fragments

Román Gajardo

Definición

- Podemos definir a un fragmento como una porción de interface de usuario o vista que se integra en un activity.
- Por lo tanto, tendremos la posibilidad de combinar múltiples fragmentos en una sola actividad o incluso, reutilizar fragmentos en otras actividades.

Ciclo de Vida



- A la hora de crear un fragmento se deben implementar al menos tres métodos
- **onCreate():** El sistema llama al método a la hora de crear el fragmento.
- **onCreateView():** El sistema llama al método a la hora de crear una interface de usuario o vista.
- **onPause():** El sistema llama al método en el momento de que el usuario abandone el fragmento.

Crear Fragment

```
public class FragmentUNO extends Fragment {

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
            Bundle savedInstanceState) {

        return inflater.inflate(R.layout.fragment_uno, container, false);
    }

}
```

- Para crear un fragmento debemos extender de la clase Fragment y sobreescribir el método onCreateView, en el que retorna la vista de dicho fragmento.

Crear Fragment

- LayoutInflater: utilizado para inflar el layout del fragmento.

- ViewGroup: es el padre en donde se va a insertar el layout del fragmento

- Bundle: utilizado para recuperar datos de instancia anterior a nuestro fragment.

Agregar Fragment a un Activity

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <fragment

        android:name="com.datohosting.fragments.FRAGMENTOS.FragmentUNO"
        android:id="@+id/fragment_uno"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>

</LinearLayout>
```

- Se debe crear un elemento fragment y especificar a través del atributo android:name la ubicación de nuestro fragmento.

Agregar Fragment mediante la Programación

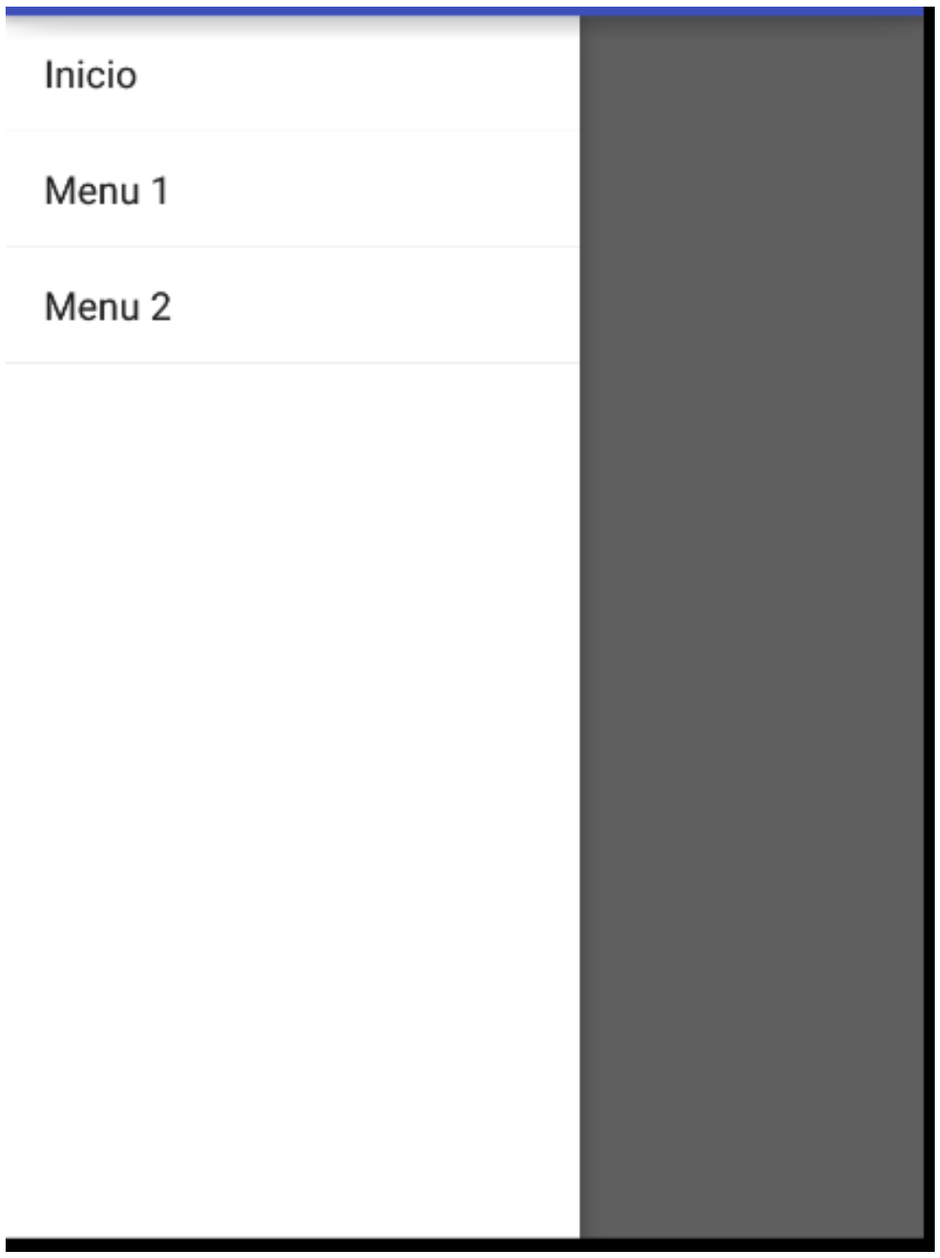
```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    setContentView(R.layout.mostrar_fragment_dos);

    FragmentManager FM = getSupportFragmentManager();
    FragmentTransaction FT = FM.beginTransaction();

    Fragment fragment = new FragmentUNO();
    FT.add(R.id.fragment_container, fragment);
    FT.commit();
}
```

- Trabajar con la librería de compatibilidad v4.

- replace

Ejercicio

DrawerLayout



- Es un tipo de layout que permite generar un menú lateral
- Debe soportar la version 4 (para móviles nuevos y antiguos)
- trabaja con Fragmentos generados a nivel de código

Estructura

```xml
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_height="match_parent"
    android:layout_width="match_parent"
    android:id="@+id/contenedor"
    >


    <FrameLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/contenedorFragment"
        ></FrameLayout>


    <ListView
        android:layout_width="240dp"
        android:layout_height="match_parent"
        android:id="@+id/menu"
        android:background="#fff"
        android:choiceMode="singleChoice"
        android:layout_gravity="start"
        >
    </ListView>

</android.support.v4.widget.DrawerLayout>
```

- **FrameLayout:** layout en el cual se cargan los fragmentos

- **ListView**: control que carga el menu de opciones.

Programación Main Activity

- Atributos

```
String opciones[];
ListView listView;
```

- onCreate

```
opciones = new String[]{"Inicio", "Menu 1", "Menu 2"};
listView = (ListView) findViewById(R.id.menu);
```

```
listView.setAdapter(new ArrayAdapter<String>(this,
                                        android.R.layout.simple_list_item_1, opciones));
```

```
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {



    }
});
```

Demo

ActionBarDrawerToggle

- Es un elemento que se implementa sobre la action bar para abrir y cerrar un Navigation Drawer con el icono de la aplicación.
- Este componente puede accionar el Navigation Drawer debido a que implementa la escucha DrawerListener en su definición.

ActionBarDrawerToggle

```
drawerToggle = new ActionBarDrawerToggle(
                    this,
                    drawerLayout,
                    R.drawable.ic_drawer,
                    R.string.drawer_open,
                    R.string.drawer_close
                ) {
    ...
```

- param 1: contexto donde se ejecuta
- param 2: el Drawer Layout con el que se relaciona
- param 3: su ícono
- param 4 y 5: strings de accesibilidad que contienen información de apertura y cierre del Drawer

ActionBarDrawerToggle

```
drawerToggle = new ActionBarDrawerToggle(
                this,
                drawerLayout,
                R.drawable.ic_drawer,
                R.string.drawer_open,
                R.string.drawer_close
                ) {
            public void onDrawerClosed(View view) {
    //Cambiar título por nombre del item en la lista
                getActionBar().setTitle(itemTitle);
            }

            public void onDrawerOpened(View drawerView) {
    //Cambiar título por nombre de la actividad
                getActionBar().setTitle(activityTitle);
            }
        };
        //Seteamos la escucha
        drawerLayout.setDrawerListener(drawerToggle);
```

- Se puede implementar eventos de apertura y cierre del Navigation Drawer

ActionBarDrawerToggle

```
@Override
  public boolean onOptionsItemSelected(MenuItem item) {

      if (drawerToggle.onOptionsItemSelected(item)) {
 // Toma los eventos de selección del toggle aquí
          return true;
      }

...//Manejo de los action buttons

      return super.onOptionsItemSelected(item);
  }
```

- Configuración del drawerToggle cuando es presionado

Demo