

Evidencia día 1 semana 14

Fragments y ActionBar

Un Fragment representa un comportamiento o una parte de la interfaz de usuario en una `FragmentActivity`. Puedes combinar varios fragmentos en una sola actividad para crear una IU multipanel y volver a usar un fragmento en diferentes actividades. Puedes pensar en un fragmento como una sección modular de una actividad que tiene un ciclo de vida propio, que recibe sus propios eventos de entrada y que puedes agregar o quitar mientras la actividad se esté ejecutando (algo así como una "subactividad" que puedes volver a usar en diferentes actividades).

Un fragmento siempre debe estar alojado en una actividad y el ciclo de vida del fragmento se ve afectado directamente por el ciclo de vida de la actividad anfitriona. Por ejemplo, cuando la actividad está pausada, también lo están todos sus fragmentos, y cuando la actividad se destruye, lo mismo ocurre con todos los fragmentos. Sin embargo, mientras una actividad se está ejecutando (está en el estado del ciclo de vida reanudada), puedes manipular cada fragmento de forma independiente, por ejemplo, para agregarlo o quitarlo. Cuando realizas una transacción de fragmentos como esta, también puedes agregarlos a una pila de actividades administrada por la actividad; cada entrada de la pila de actividades en la actividad es un registro de la transacción de fragmentos realizada. La pila de actividades le permite al usuario invertir una transacción de fragmentos (navegar hacia atrás) al presionar el botón Atrás.

Cuando se agrega un fragmento como parte de la presentación de la actividad, se encuentra en `ViewGroup` dentro de la jerarquía de vistas de la actividad y el fragmento define su propio diseño de vistas. Puedes insertar un fragmento en el diseño de la actividad declarando el fragmento en el archivo de diseño, como elemento `<fragment>`, o desde el código de tu aplicación agregándolo a un archivo existente `ViewGroup`.

Ejercicio

Realizar Fragments y Action Bar según ejemplo visto en clases.

Se importan los vectores (Íconos) que usaremos para identificar los botones del menú actionBar y que corresponden a cada fragmento de nuestra Main Activity.

Project/res/drawable → New Vector asset // Se edita algunas propiedades (Color, Nombre etc) y se guarda en la carpeta específica en Drawable.

Se crea un menú → Click derecho en Res, New android Resource file, Archivo tipo Menú. Luego se crean 3 fragments dentro de la carpeta App para cada uno de los componentes que van a estar presentes en la Action Bar. Cada `TextView` de cada fragmet se cambia a uno que indique a que se le dio click (Un auto, un avión o un barco). Con los fragments creados, se da la interfaz al menú en el archivo `menu.xml` de diseño.

Menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu
    xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/inicio"
        android:icon="@drawable/casablanca"
        android:title="Inicio" />
    <item
        android:id="@+id/buscar"
        android:icon="@drawable/buscarblanco"
        android:title="Buscar"/>

    <item
        android:id="@+id/perfil"
        android:icon="@drawable/perfilblanco"
        android:title="perfil" />
</menu>
```

ActivityMain.xml

```
<?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"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/navigationbuttonview"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:menu="@menu/menu"
        android:background="#07bbf0"
        app:itemIconTint="#ffff"
        app:itemTextColor="#ffff"

    />

    <FrameLayout
        android:id="@+id/fragment_container"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        app:layout_constraintTop_toBottomOf="@id/navigationbuttonview"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toEndOf="parent"
```

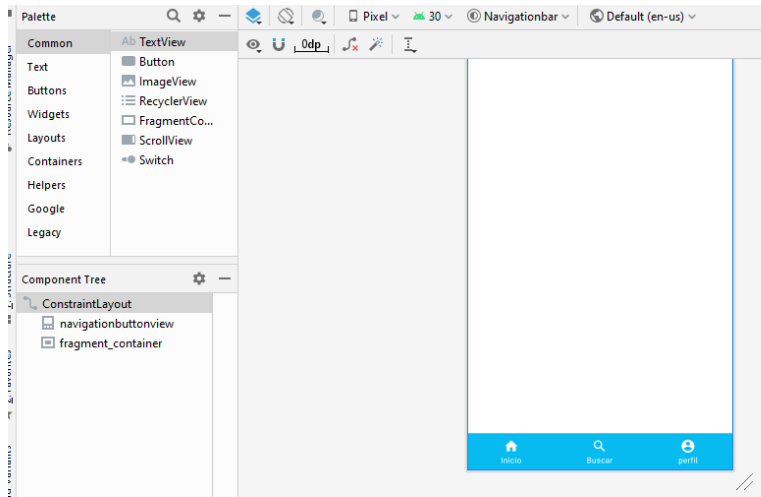
```

        app:layout_constraintTop_toTopOf ="parent"
    />
</androidx.constraintlayout.widget.ConstraintLayout>

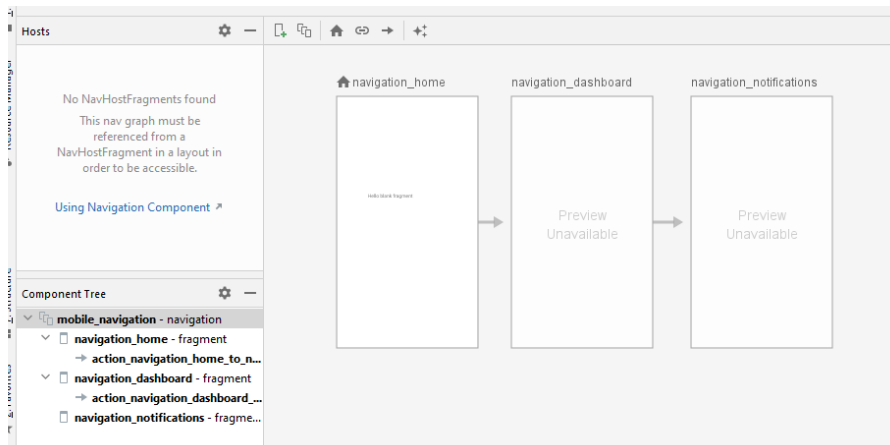
```

Diseño

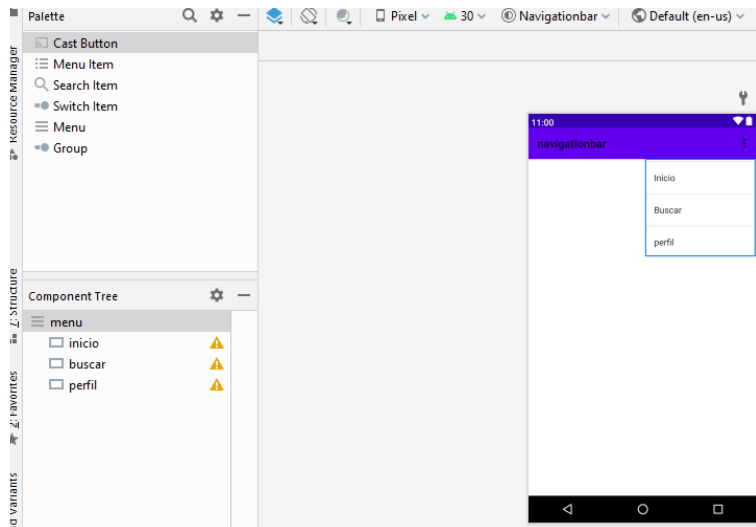
ActivityMain



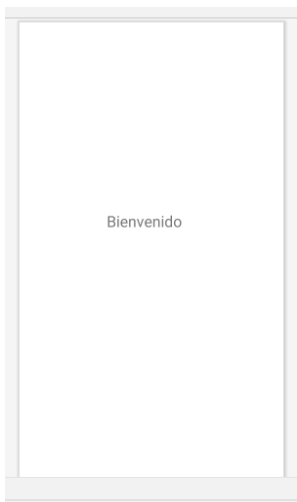
Mobile Navigation



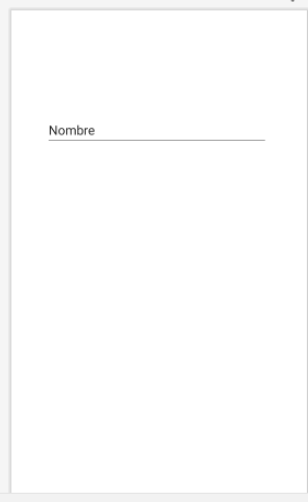
Menú



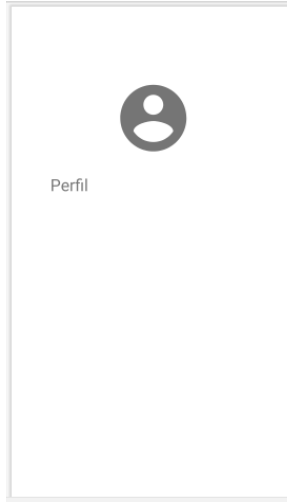
Fragment Home



Fragment Search



Fragment Profile



Código ActivityMain

```
package com.example.navigationBar
```

```
import androidx.appcompat.app.AppCompatActivity
```

```
import android.os.Bundle
```

```
import androidx.fragment.app.Fragment
```

```
import com.google.android.material.bottomnavigation.BottomNavigationView
```

```
class MainActivity : AppCompatActivity() {
```

```
    val iniciofragment = HomeFragment()
```

```
    val buscarfragment = SearchFragment()
```

```
    val perfilfragment = ProfileFragment()
```

```
    override fun onCreate(savedInstanceState: Bundle?) {
```

```
        super.onCreate(savedInstanceState)
```

```
        setContentView(R.layout.activity_main)
```

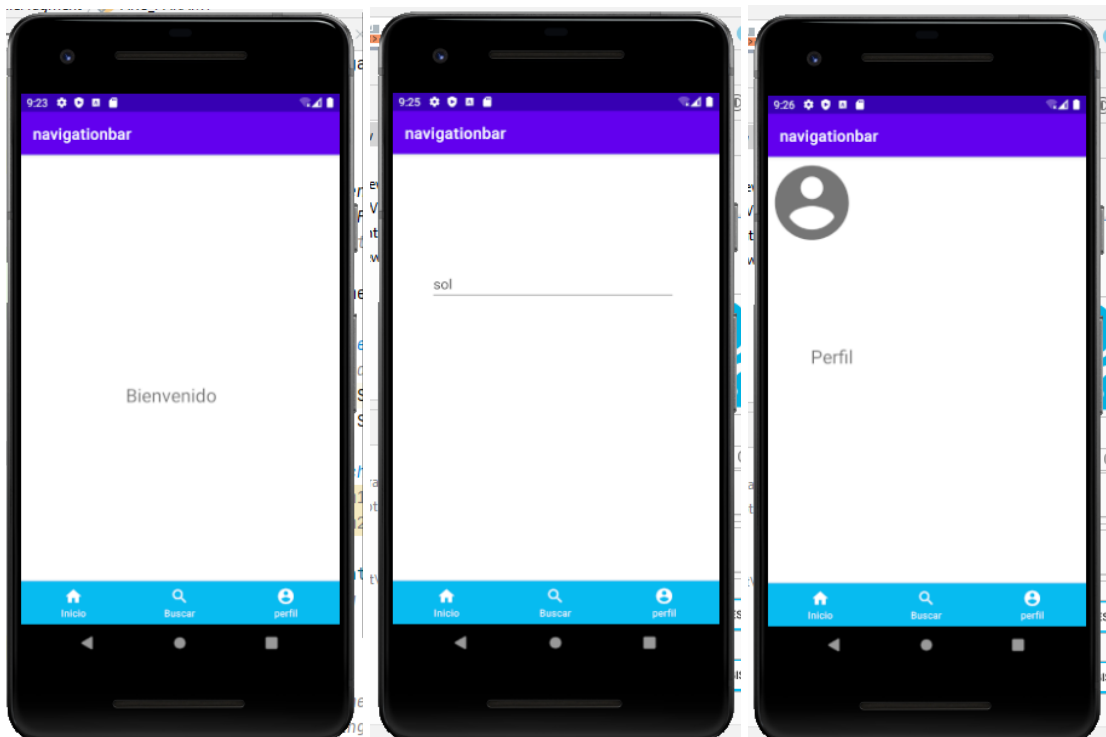
```

val bottom_navigation= findViewById<BottomNavigationView>(R.id.navigationbuttonview)

reemplazar(iniciofragment)
bottom_navigation.setOnItemSelectedListener {
    when (it.itemId) {
        R.id.inicio -> reemplazar(iniciofragment)
        R.id.buscar -> reemplazar(buscarfragment)
        R.id.perfil -> reemplazar(perfilfragment)
    }
    true
}
}
}
fun reemplazar(fragment: Fragment) {
    if (fragment!=null){
        val transaccion = supportFragmentManager.beginTransaction()
        transaccion.replace (R.id.fragment_container, fragment)
        transaccion.commit()
    }
}
}
}

```

Emulador



Reflexión: La clase de hoy estuvo más interesante y revisamos cómo se comportan distintos fragmentos en una misma actividad.