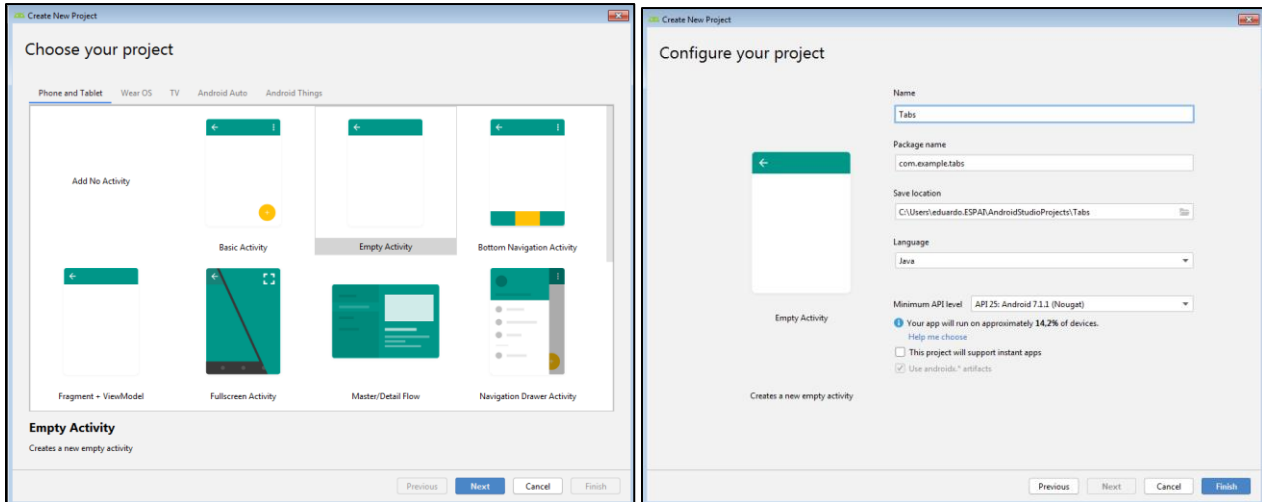


PRACTICA 6: USO DE FRAGMENTTABHOST

Parte I: Uso de FragmentTabHost

Paso 1. Crea un nuevo proyecto y llámalo Tabs:



Paso 2. Reemplaza el código de `activity_main.xml` por el siguiente:

```
activity_main.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.fragment.app.FragmentTabHost
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      android:id="@android:id/tabhost"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent" >
7      <LinearLayout
8          android:layout_width="match_parent"
9          android:layout_height="match_parent"
10         android:orientation="vertical" >
11         <TabWidget
12             android:id="@android:id/tabs"
13             android:layout_width="match_parent"
14             android:layout_height="wrap_content"
15             android:layout_weight="0"
16             android:orientation="horizontal" />
17         <FrameLayout
18             android:id="@android:id/tabcontent"
19             android:layout_width="match_parent"
20             android:layout_height="0dp"
21             android:layout_weight="1" />
22     </LinearLayout>
23 </androidx.fragment.app.FragmentTabHost>
```

La primera cosa extraña de este código es el nombre de la primera etiqueta. En lugar de indicar simplemente `FragmentTabHost`, se ha indicado el nombre completo de dominio. Cada vez que usemos una vista que no sea del sistema (por ejemplo creada por nosotros o creada en una librería como en este caso) tendremos que indicar la clase donde se crea con su nombre completamente cualificado. Es decir, el nombre de la clase precedida de su paquete.

Como puedes observar un `FragmentTabHost` es el nodo raíz del diseño, que contiene dos elementos combinados por medio de un `LinearLayout`. El primero es un `TabWidget` para la visualización de las pestañas y el segundo es un `FrameLayout` para mostrar el contenido asociado de cada lengüeta. El número de lengüetas y su contenido se indicará por código.

Paso 3. Abre el fichero *MainActivity.java* y reemplaza el código por el siguiente.

```
1 package com.example.tabs;
2
3 import android.os.Bundle;
4
5 import androidx.fragment.app.FragmentActivity;
6 import androidx.fragment.app.FragmentTabHost;
7
8 public class MainActivity extends FragmentActivity {
9     private FragmentTabHost tabHost;
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_main);
15         tabHost= (FragmentTabHost) findViewById(android.R.id.tabhost);
16         tabHost.setup( context: this, getSupportFragmentManager(), android.R.id.tabcontent);
17         tabHost.addTab(tabHost.newTabSpec( tag: "tab1").setIndicator("Pestaña 1"), Tab1.class, args: null);
18         tabHost.addTab(tabHost.newTabSpec( tag: "tab2").setIndicator("Pestaña 2"), Tab2.class, args: null);
19         tabHost.addTab(tabHost.newTabSpec( tag: "tab3").setIndicator("Pestaña 3"), Tab3.class, args: null);
20     }
21 }
22 }
```

Observa como la clase creada extiende de *FragmentActivity* en lugar de *Activity*. Esto permitirá que la actividad trabaje con fragments; en concreto, vamos a crear un fragment para cada lengüeta. Se han añadido varias líneas en el método *onCreate()*. Empezamos inicializando la variable *tabHost*, luego se utiliza el método *setup()* para configurarla. Para ello indicamos el contexto, manejador de fragments y donde se mostrarán los fragments.

Cada una de las siguientes tres líneas introduce una nueva lengüeta usando el método *addTab()*. Se indican tres parámetros: un objeto *TabSpec*, una clase con el fragment a visualizar en la lengüeta y un *Bundle* por si queremos pasar información a la lengüeta. El método *newTabSpec()* crea una nueva lengüeta en un *TabHost*. Se le pasa como parámetro un *String*, que se utiliza como identificador y devuelve el objeto de tipo *TabSpec* creado.

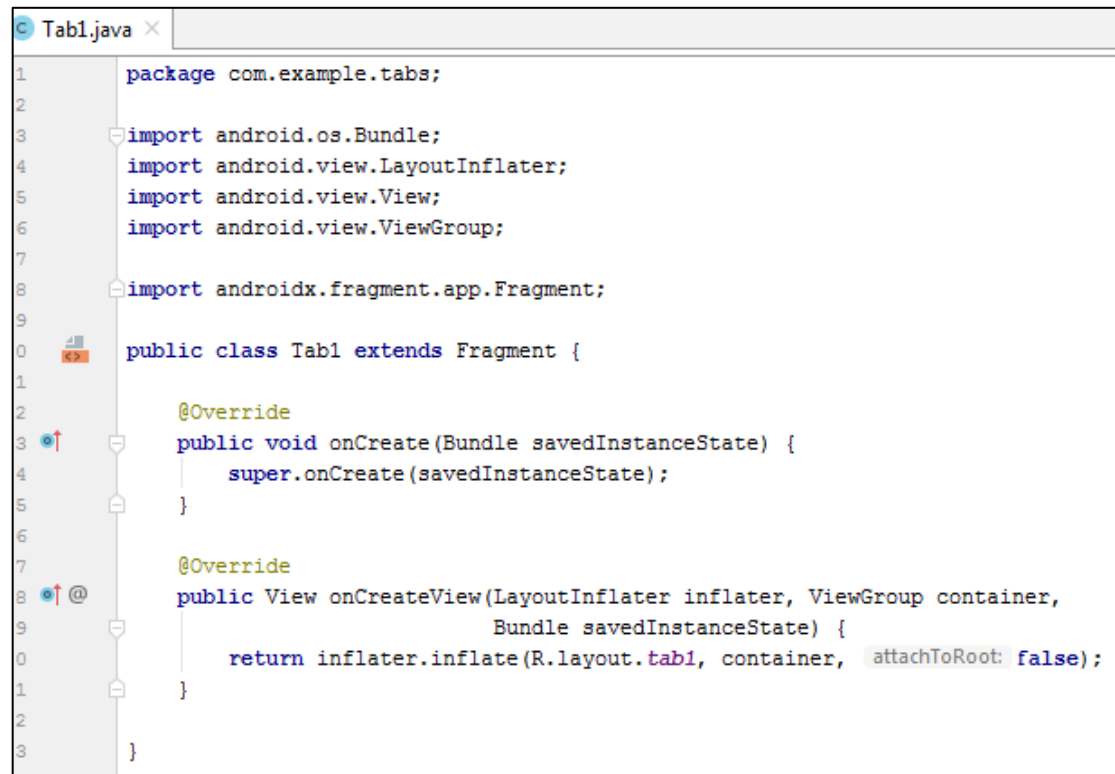
Nota sobre Java: Dado que el método *newTabSpec()* devuelve un objeto de tipo *TabSpec*, tras la llamada, podemos llamar al método *setIndicator()*, que nos permitirá introducir un descriptor en la pestaña recién creada.

NOTA: También podremos asignar iconos a las lengüetas con el método *setIndicator()*. En el capítulo siguiente se estudiarán los iconos disponibles en el sistema y cómo crear nuevos icono. En las últimas versiones de Android solo podemos visualizar un texto o un icono. Para ver el icono introduce un texto vacío.

Paso 4. Crea un nuevo layout y llámalo *tab1.xml*:

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <LinearLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     android:layout_width="match_parent"
5     android:layout_height="match_parent"
6     android:orientation="vertical" >
7     <TextView
8         android:id="@+id/text"
9         android:layout_width="match_parent"
10        android:layout_height="match_parent"
11        android:gravity="center_vertical|center_horizontal"
12        android:text="Pestaña 1"
13        android:textAppearance="?android:attr/textAppearanceMedium" />
14 </LinearLayout>
```

Paso 5. Crea una nueva clase con *Tab1.java*:

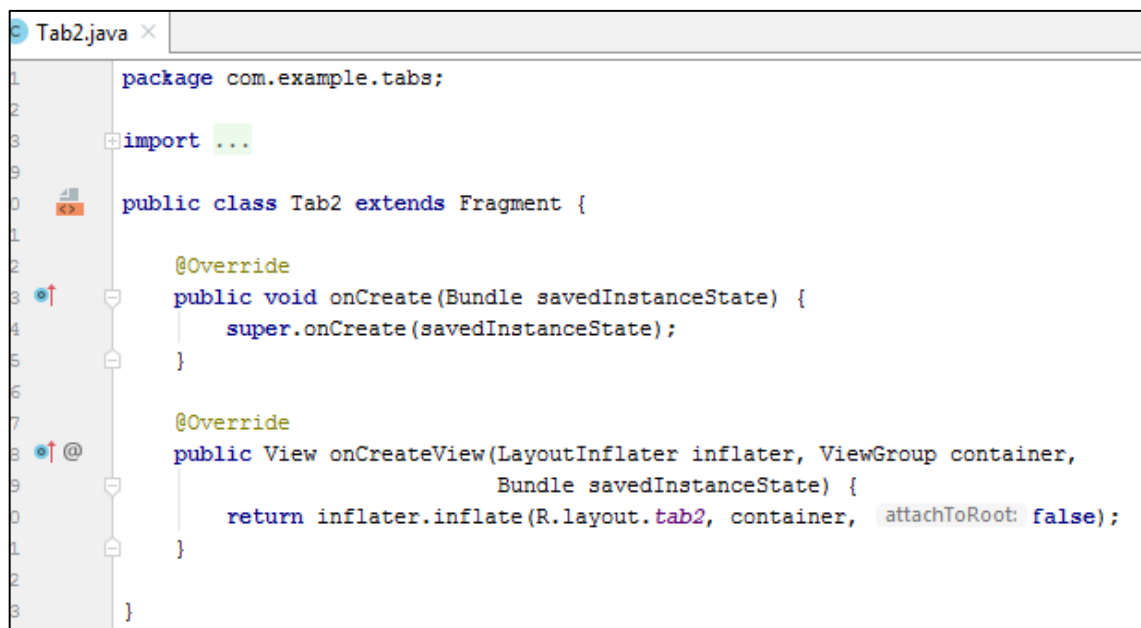
A screenshot of an IDE showing the code for Tab1.java. The code is as follows:

```
1 package com.example.tabs;
2
3 import android.os.Bundle;
4 import android.view.LayoutInflater;
5 import android.view.View;
6 import android.view.ViewGroup;
7
8 import androidx.fragment.app.Fragment;
9
10 public class Tab1 extends Fragment {
11
12     @Override
13     public void onCreate(Bundle savedInstanceState) {
14         super.onCreate(savedInstanceState);
15     }
16
17     @Override
18     public View onCreateView(LayoutInflater inflater, ViewGroup container,
19                             Bundle savedInstanceState) {
20         return inflater.inflate(R.layout.tab1, container, attachToRoot: false);
21     }
22 }
```

Nota sobre Java: Pulsa *Alt-Intro* en Android Studio o *Ctrl-Shift-O* en Eclipse para que automáticamente se añadan los paquetes que faltan. Si la clase *Fragment* se encuentra en más de un paquete, selecciona *androidx.fragment.app.Fragment*.

Un fragment se crea de forma muy parecida a una actividad. También dispone del método *onCreate()*, aunque en este ejemplo no se introduce código. Un fragment también tiene asociada una vista, aunque a diferencia de una actividad, no se asocia en el método *onCreate()*, sino que dispone de un método especial para esta tarea: *onCreateView()*.

Paso 6. Repite los dos pasos anteriores para crear *tab2.xml* y *Tab2.java*.

A screenshot of an IDE showing the code for Tab2.java. The code is as follows:

```
1 package com.example.tabs;
2
3 import ...
4
5 public class Tab2 extends Fragment {
6
7     @Override
8     public void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10     }
11
12     @Override
13     public View onCreateView(LayoutInflater inflater, ViewGroup container,
14                             Bundle savedInstanceState) {
15         return inflater.inflate(R.layout.tab2, container, attachToRoot: false);
16     }
17 }
```

```

tab2.xml x
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:orientation="vertical" >
7      <TextView
8          android:id="@+id/text"
9          android:layout_width="match_parent"
10         android:layout_height="match_parent"
11         android:gravity="center_vertical|center_horizontal"
12         android:text="Pestaña 2"
13         android:textAppearance="?android:attr/textAppearanceMedium" />
14  </LinearLayout>

```

Paso 7. Repite de nuevo para crear el layout tab3.xml y la clase Tab3.java.

```

Tab3.java x
package com.example.tabs;

import ...

public class Tab3 extends Fragment {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        return inflater.inflate(R.layout.tab3, container, attachToRoot: false);
    }

}

```

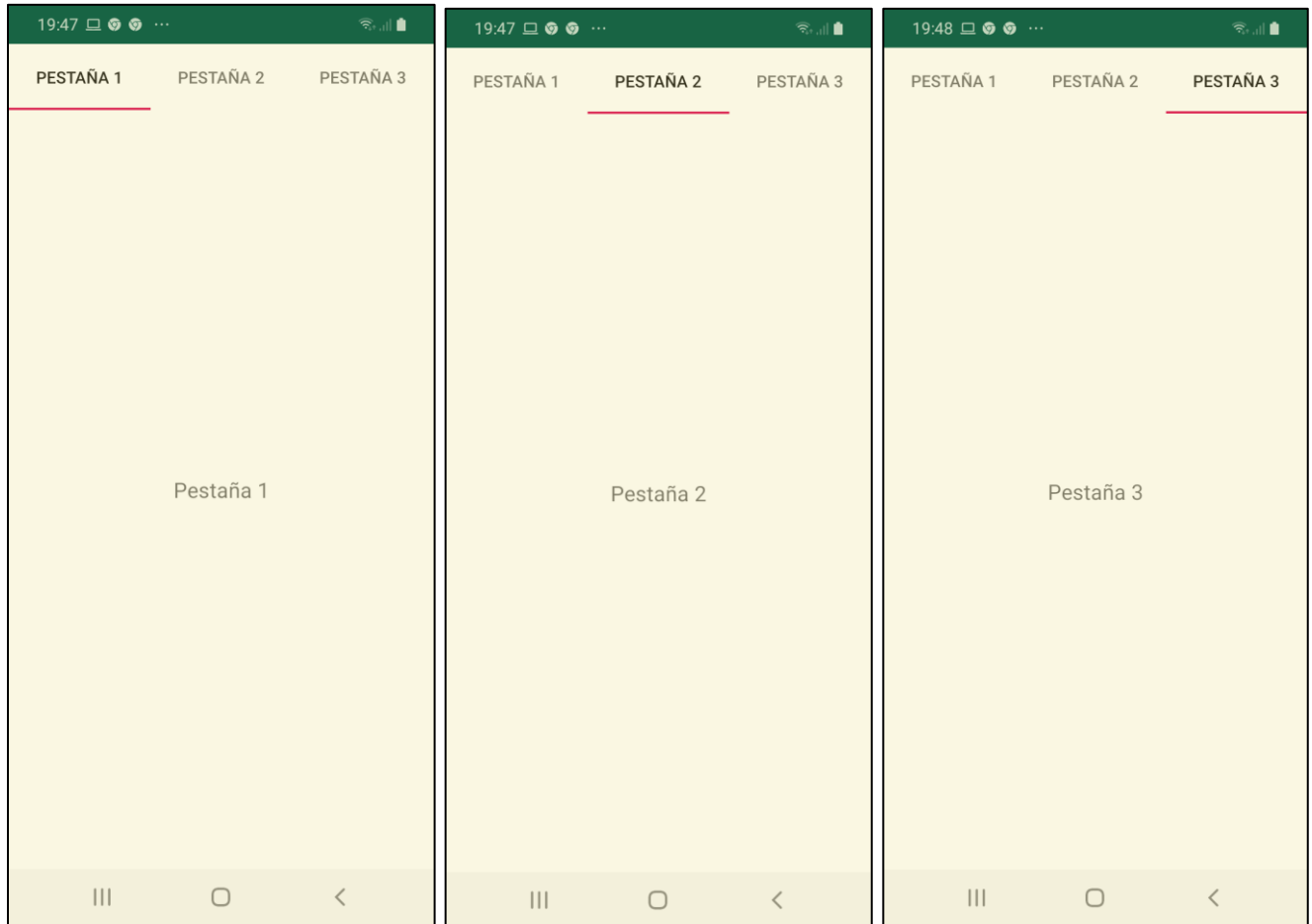
```

tab3.xml x
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:orientation="vertical" >
7      <TextView
8          android:id="@+id/text"
9          android:layout_width="match_parent"
10         android:layout_height="match_parent"
11         android:gravity="center_vertical|center_horizontal"
12         android:text="Pestaña 3"
13         android:textAppearance="?android:attr/textAppearanceMedium" />
14  </LinearLayout>

```

Paso 8. Modifica estos ficheros para que cada layout sea diferente y para que cada fragment visualice el layout correspondiente.

Paso 9. Ejecuta el proyecto y verifica el resultado.



NOTA: Si en uno de los layouts asignados a un fragment has utilizado el atributo `onClick`, el método indicado ha de ser introducido dentro de la actividad. Si lo introduces dentro del fragment no será reconocido.