**INGENIERIA DE SISTEMAS**

****

**Crear e Implementar una librería en**

**Android Studio**

**NOMBRE:** RAYA SALAZAR ROLANDO JHONATAN

**Docente:** Ing. William Barra

**INDICE**

Contenido

[1. Introducción 3](#_Toc43830280)

[1.1. Modulo Android Studio 3](#_Toc43830281)

[1.2. Planteamiento del problema 3](#_Toc43830282)

[1.2.1. Descripción del problema 3](#_Toc43830283)

[1.3. Como crear un modulo de biblioteca 3](#_Toc43830284)

[1.3.1. Nuevo Modulo 4](#_Toc43830285)

[1.3.2 Crear libreria 4](#_Toc43830286)

[1.4. Implementación de la Liberia a un proyecto Android en blanco 6](#_Toc43830287)

[1.5. Una ves implementedado 9](#_Toc43830288)

[1.6. Iniciamos la app con la librería 10](#_Toc43830289)

[1.7. Conclusiones 14](#_Toc43830290)

# 1. Introducción

Una biblioteca de Android tiene la misma estructura que un módulo de app de Android. Puede incluir todo lo que se necesita para compilar una app, como el código fuente, archivos de recursos y un manifiesto de Android. Sin embargo, en lugar de compilarse en un APK que se ejecute en un dispositivo, una biblioteca de Android se compila en un archivo Android ARchive (AAR) que puedes usar como dependencia para un módulo de app de Android. A diferencia de los archivos JAR, los archivos AAR pueden contener recursos de Android y un archivo de manifiesto, que te permite incluir recursos compartidos, como diseños y elementos de diseño, además de clases y métodos de Java.

## Modulo Android Studio

Un módulo de biblioteca es útil en las siguientes situaciones:

* Cuando compilas varias apps que comparten algunos componentes, como actividades, servicios o diseños de IU
* Cuando compilas una app que existe en diferentes variantes del APK, como una versión gratuita o paga, y necesitas los principales componentes en ambas variantes

## 1.2. Planteamiento del problema

### 1.2.1. Descripción del problema

Con la finalidad de implementar partes importantes de otros proyectos a nuestro proyecto Android Studio crearemos la librería de un OnBoarding para poder implementarlo en u n proyecto nuevo y no tener que realizar el OnBoarding desde cero.

Esta librería nos facilita el trabajo disminuyendo en tiempo de trabajo en implementar un OnBoarding o cual quiera que sea la librería. Nos Ayuda a llamar de manera fácil a algo especifico que necesitamos usar en nuestra app de Android Studio.

## 1.3. Como crear un modulo de biblioteca

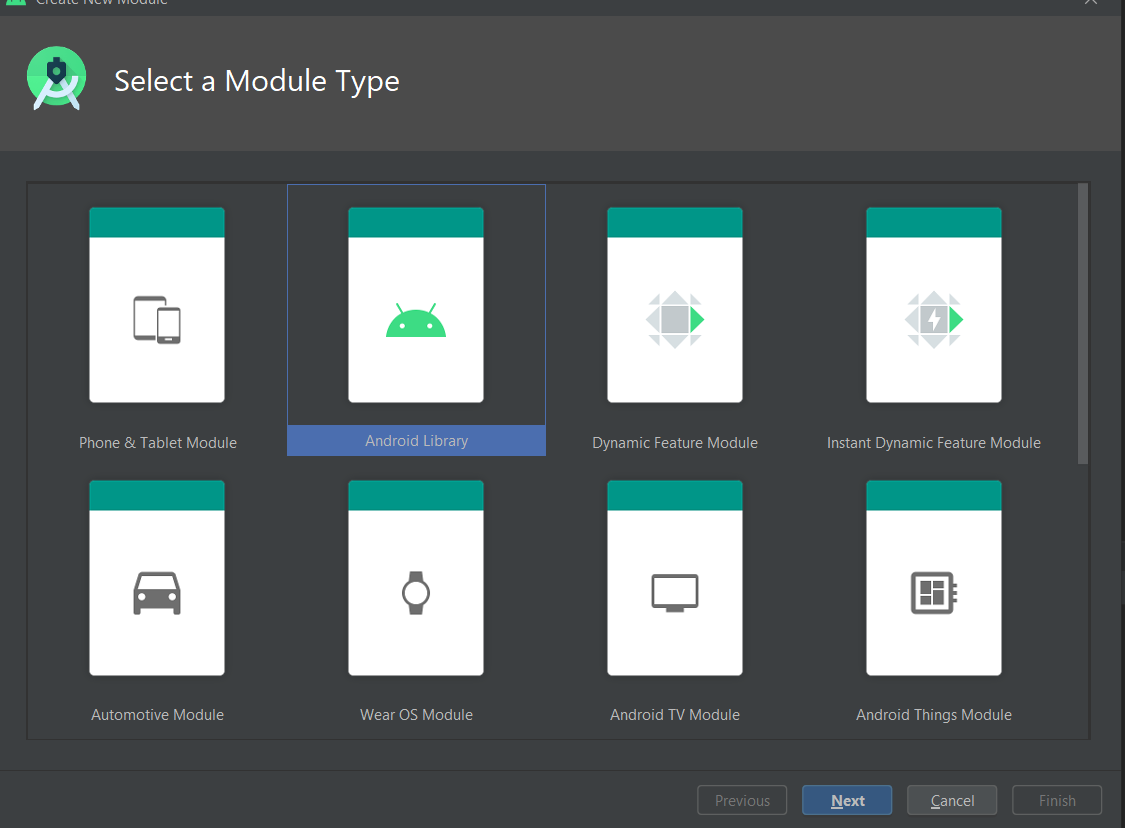
Para crear un módulo de biblioteca en tu proyecto, haz lo siguiente:

### 1.3.1. Nuevo Modulo

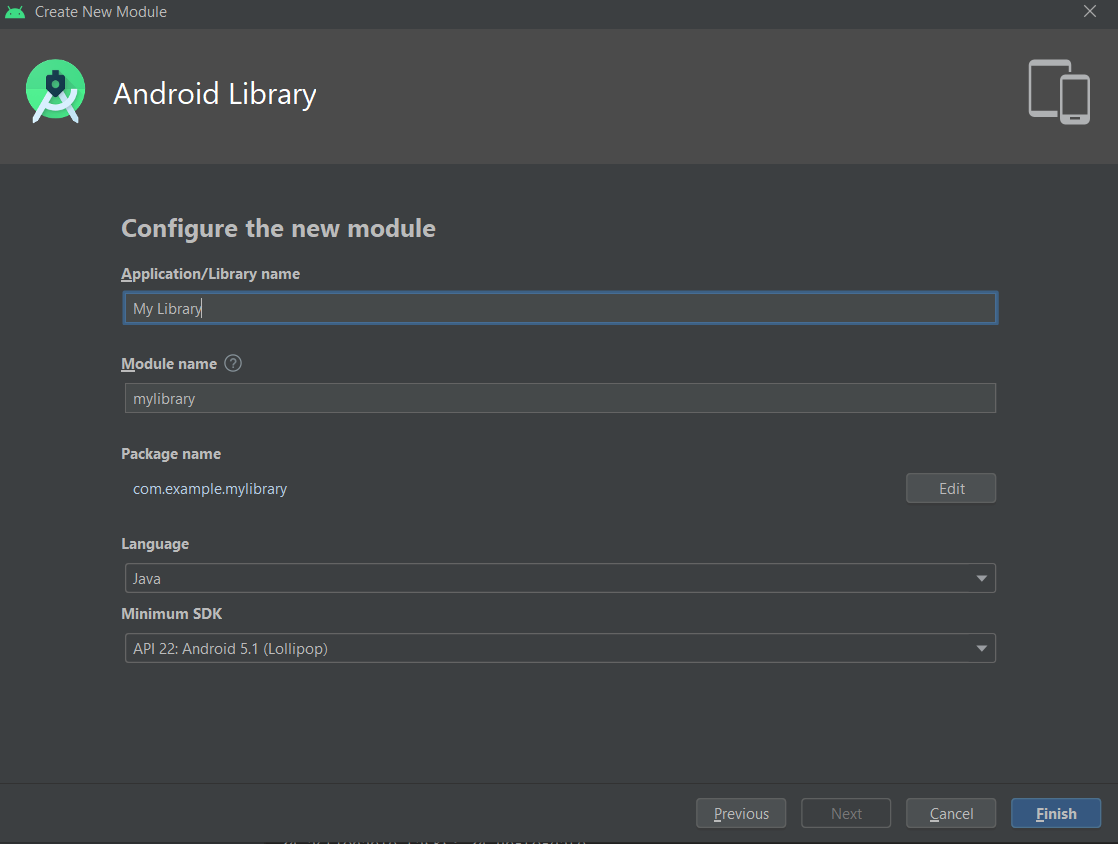
Haz clic en **File > New > New Module**.

### 1.3.2 Crear libreria

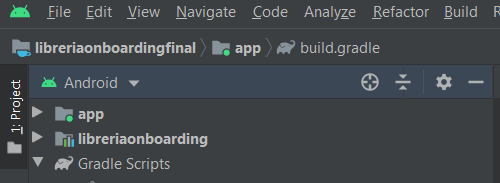
En la ventana **Create New Module** que aparece, haz clic en **Android Library** y, luego, en **Next**



* Otorga un nombre a la biblioteca y selecciona una versión mínima del SDK para el código de la biblioteca. Luego, haz clic en **Finish**.



* Guarda el archivo y haz clic en **File > Sync Project with Gradle Files**.
* Asegúrate de que se muestre la biblioteca en la parte superior del archivo settings. gralde, como se muestra aquí para una biblioteca denominada "my-library-module":



Abre el archivo build. gradle del módulo de la app y agrega una nueva línea al bloque dependencies como se muestra en el siguiente fragmento:



* Haz clic en **Sync Project with Gradle Files**.

En este ejemplo anterior, la configuración de implementación agrega la biblioteca llamada my-library-module como una dependencia de compilación para todo el módulo de la app. Si, en cambio, solo quieres usar la biblioteca para variante de compilación específica, en lugar de implementación, usa buildVariantNameImplementation

## 1.4. Implementación de la Liberia a un proyecto Android en blanco

Seguimos los siguientes pasos:

//MainActivity

package com.example.pruebalib;

import androidx.appcompat.app.AppCompatActivity;

import androidx.viewpager.widget.ViewPager;

import android.os.Bundle;

import com.example.libreriaonboarding.Adapters.OnboardingAdapter;

public class MainActivity extends AppCompatActivity {

private ViewPager viewPagerContainer;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

\_inicializeComponents();

\_initializeFragmentAdapter();

}

public void \_inicializeComponents(){

viewPagerContainer = findViewById(R.id.viewPagerContainer);

}

public void \_initializeFragmentAdapter(){

OnboardingAdapter adapter = new OnboardingAdapter(getSupportFragmentManager());

viewPagerContainer.setAdapter(adapter);

}

}

//XML activity main

<androidx.viewpager.widget.ViewPager

android:id="@+id/viewPagerContainer"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

</androidx.viewpager.widget.ViewPager>

//Build gradle

allprojects {

repositories {

...

maven { url 'https://jitpack.io' }

}

}

// agregamos las dependencias

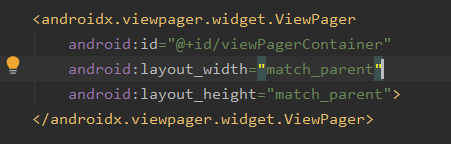
dependencies {

implementation 'com.github.jhonatanraya:libreriaonboardingfinal1:0.1'

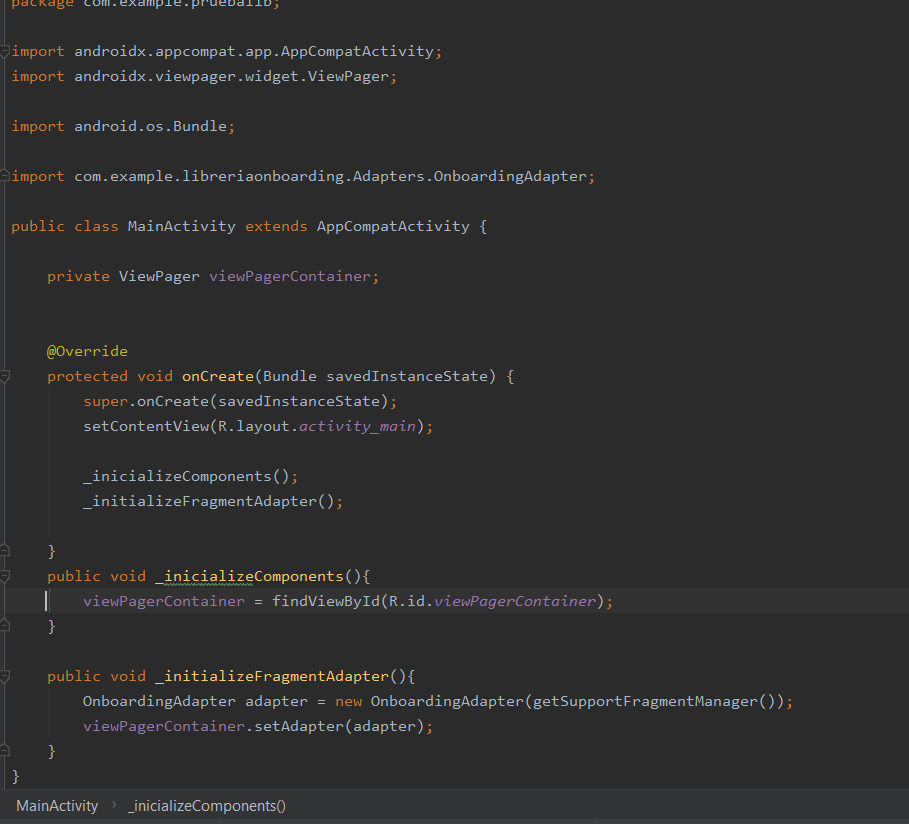
}

## 1.5. Una ves implementedado

En el XML el viewpagwer:



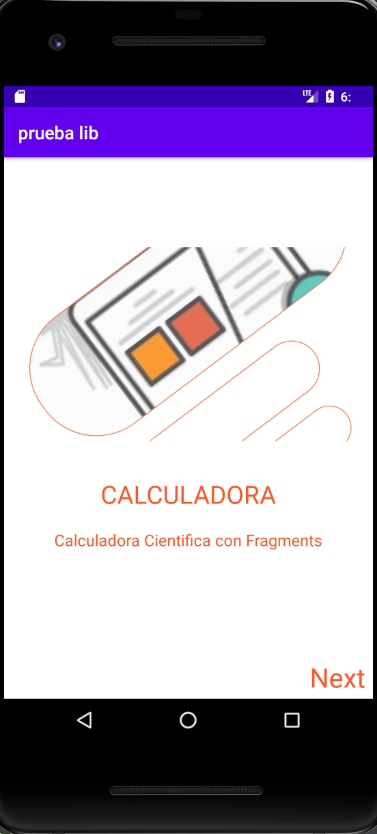
En el MainActivity inicializamos los componentes:



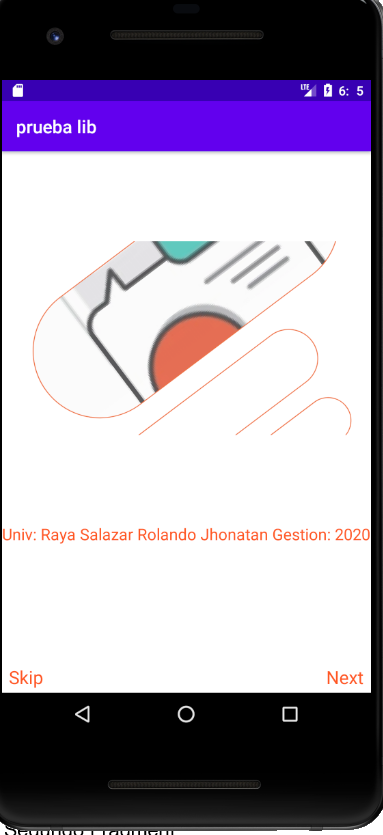
## 1.6. Iniciamos la app con la librería

Y se tendría que visualizarlos OnBoarding de la siguiente manera:

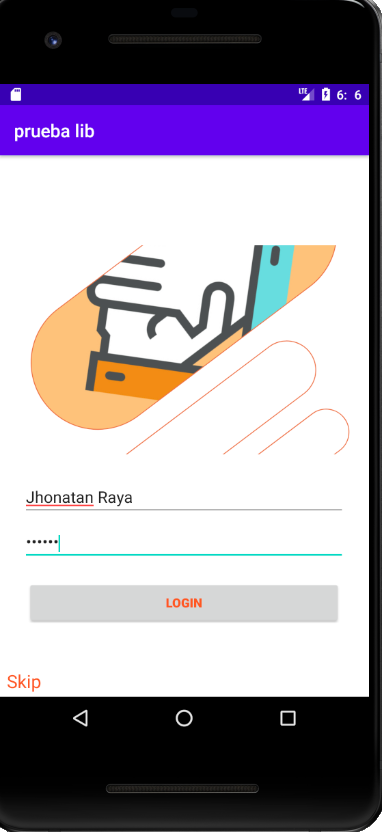
Primer fragment



Segundo Fragment:



Tercer Fragment:



## 1.7. Conclusiones

De esta manera implementamos nuestra librería de OnBoarding en un proyecto de Android Studio y esta subida en un repositorio de GitHub disponible para quien desee implementar este diseño de OnBoarding que esta completamente funcional.