

Trabajo practico N°3

Creación de SPLASH SCREEN personalizado

Alumno: Darian Hiebl

Docente: Guillermo Alfaro

Repositorio:

<https://github.com/sirnas1983/FunTelSPLASHSCREEN>

OBJETIVO

El principal objetivo de este informe es mostrar el procedimiento para la creación de una pantalla de inicio personalizada para una aplicación Android.

CONTEXTO

Se nos pide la creación de una pantalla de inicio con una duración predeterminada donde se aprecie el icono de la aplicación y opcionalmente alguna animación.

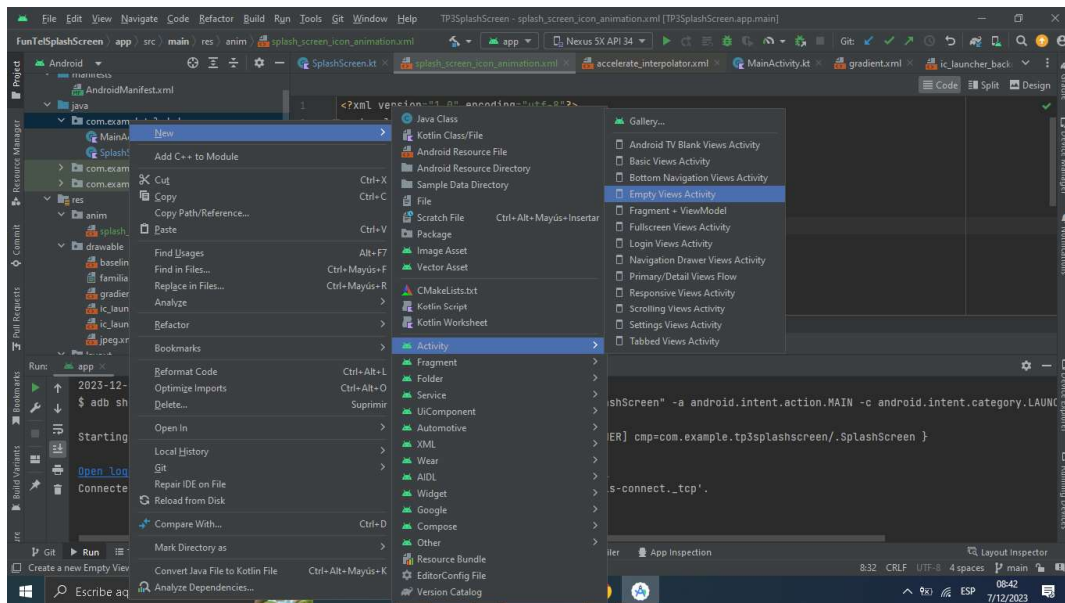
Las versiones de Android 12 traen una pantalla de inicio por defecto. Para este ejercicio vamos a deshabilitar esta función ya que la que usa por defecto Android tiene tanto la posición como el tamaño del icono fijos e inamovibles, lo cual lo hace poco flexible.

PROCEDIMIENTO

Como lo mencionamos anteriormente vamos a hacer una pantalla de inicio personalizada. El método adoptado fue el del uso de una Activity.

Paso 1) Creación de una Empty View Activity.

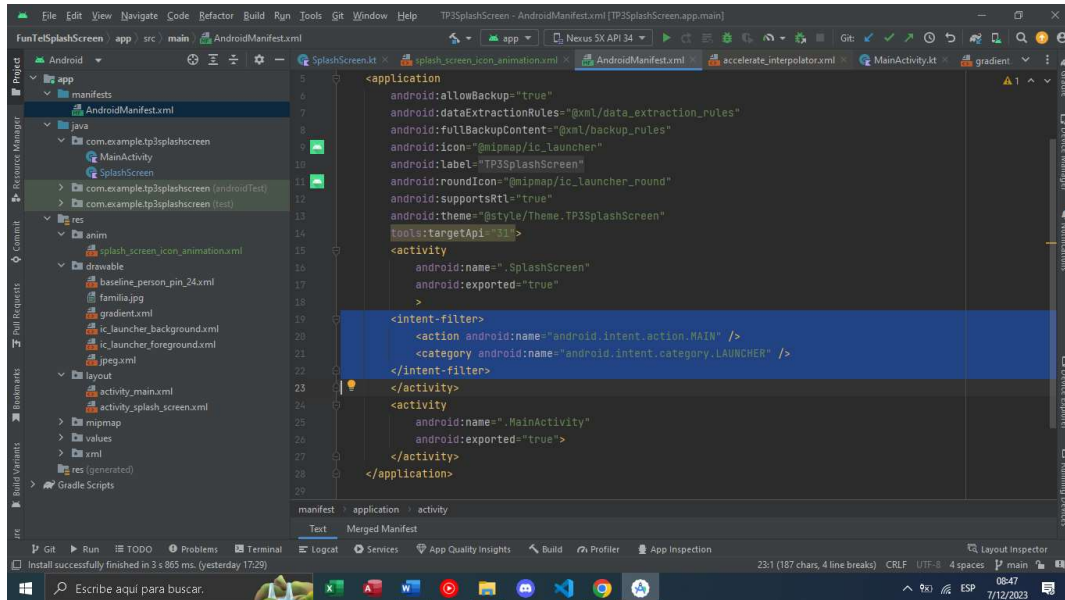
Esto se logra haciendo click con el botón derecho sobre el package de la Main Activity y seleccionando la opción New->Activity->Empty View Activity



Una vez hecho esto nombramos la activity con un nombre que sea representativo de la actividad que va a representar. En mi caso la llame SplashScreen.

Paso 2) Modificando el Manifest.xml

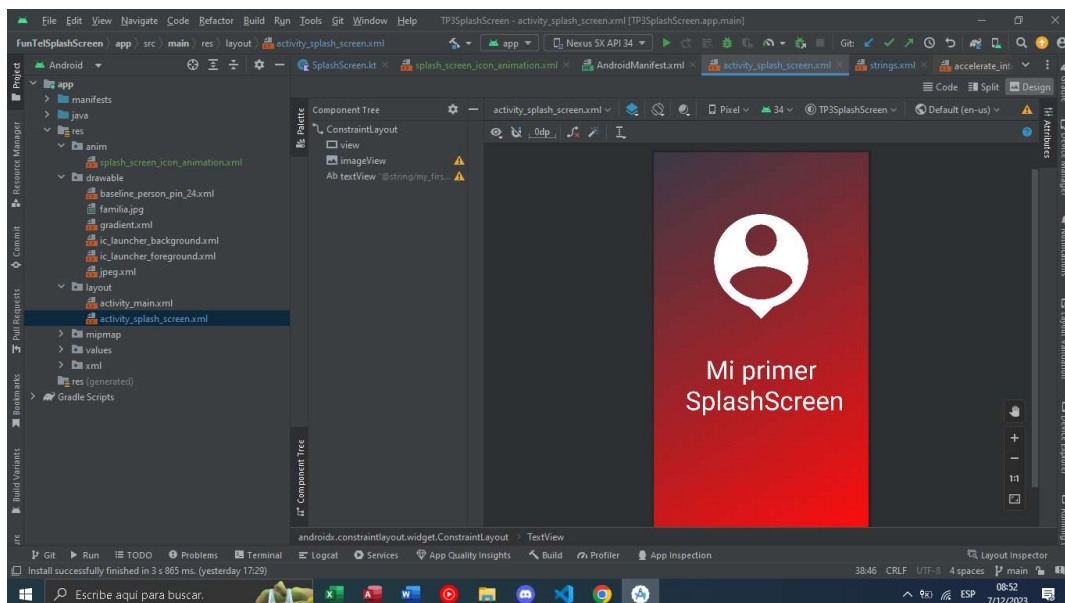
Lo que debemos hacer a continuación es indicarle a Android que la actividad de donde arrancara la aplicación es la Activity recién creada. Para ello debemos pasar el <intent> de la Main Activity a la Activity nueva dentro del manifiesto.



Lo que esta resaltado en la captura inicialmente estaba dentro de la MainActivity. Lo único que hice fue cortar y pegar en la SplashScreen.

Paso 3) Modificar el Layout de la SplashScreen

A continuación procedemos a diseñar el Layout de la SplashScreen. En mi caso hice un fondo con degrade (con un drawable) y le puse un icono que tiene Android Studio en su librería. También agregue un pequeño titulo que dice “Mi primer SplashScreen”.

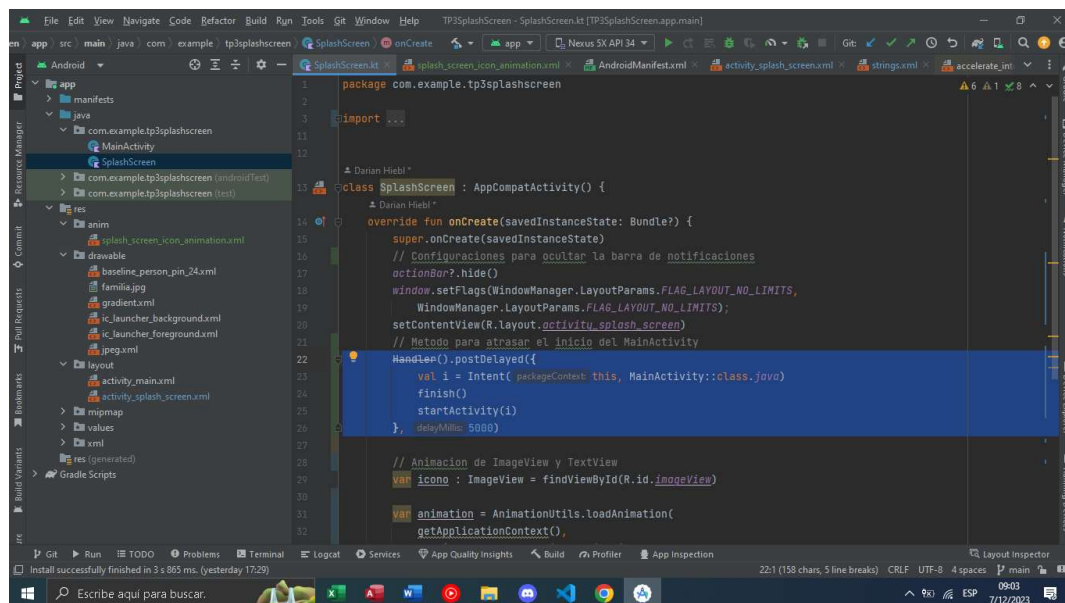


El layout consta de 3 elementos:

- 1- View sobre el cual le asigne el fondo degrade.
- 2- ImageView al cual le asigne el icono
- 3- TextView al cual le coloque el texto “Mi primer SplashScreen”

Paso 4) Inicialización del MainActivity

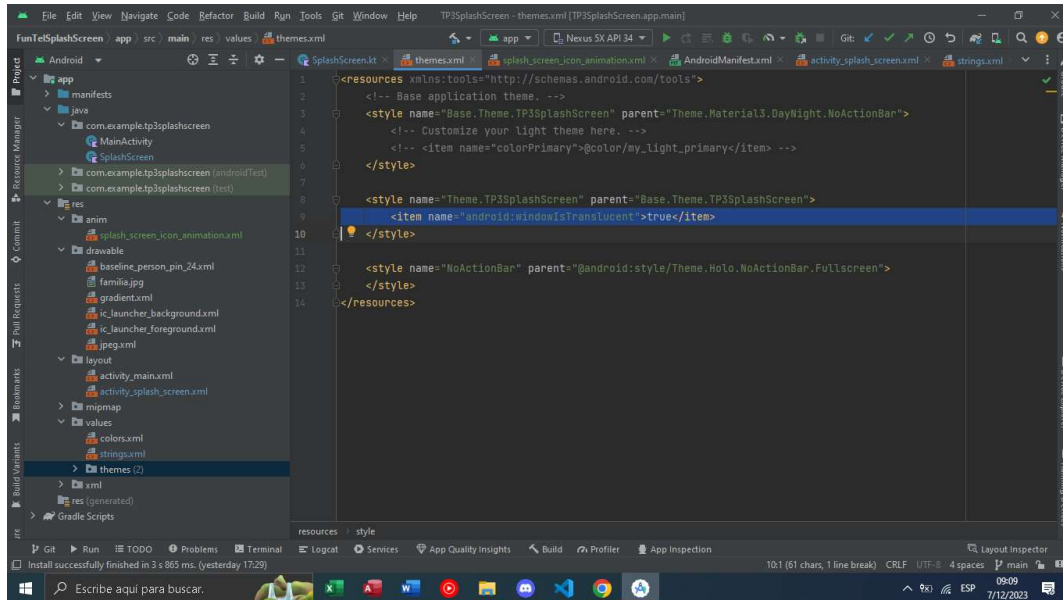
Debemos crear una ventana de tiempo entre el inicio de la SplashActivity y el MainActivity. Para lograr esto utilice la clase Handler y su método postDelayed. Esta clase permite retrasar con la cantidad de milisegundos que uno le pase como parametro cualquier acción que uno le pase. En este caso le pase la inicialización de la MainActivity con 5000 ms de delay.



```
1 package com.example.tp3splashscreen
2
3 import androidx.appcompat.app.AppCompatActivity
4
5 class SplashScreen : AppCompatActivity() {
6     override fun onCreate(savedInstanceState: Bundle?) {
7         super.onCreate(savedInstanceState)
8         // Configuraciones para ocultar la barra de notificaciones
9         actionBar?.hide()
10        window.setFlags(WindowManager.LayoutParams.FLAG_LAYOUT_NO_LIMITS,
11            WindowManager.LayoutParams.FLAG_LAYOUT_NO_LIMITS);
12        setContentView(R.layout.activity_splash_screen)
13        // Metodo para atrasar el inicio del MainActivity
14        Handler().postDelayed({
15            val i = Intent(packageContext, MainActivity::class.java)
16            finish()
17            startActivity(i)
18        }, delayMillis = 5000)
19
20        // Animacion de ImageView y TextView
21        var icono : ImageView = findViewById(R.id.imageView)
22        var animation = AnimationUtils.loadAnimation(
23            getApplicationContext(),
24            R.anim.splash_screen_icon_animation)
25    }
26}
```

Paso 5) Deshabilitar el SplashScreen por defecto

Esto en realidad no es posible, lo que se hace es volverlo totalmente transparente. Para ello, dentro del tema que se este utilizando, hay que agregar un ítem indicando que no va a ser visible.

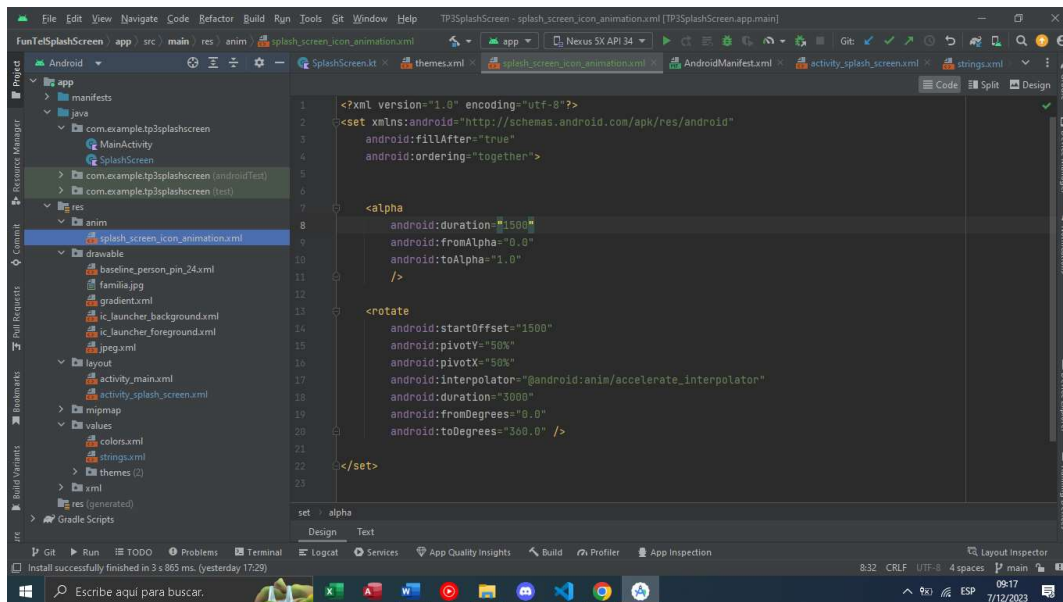


Esa línea resalta le indica a Android que la ventana de inicio por defecto sea transparente.

Paso 6) Animando la cosa

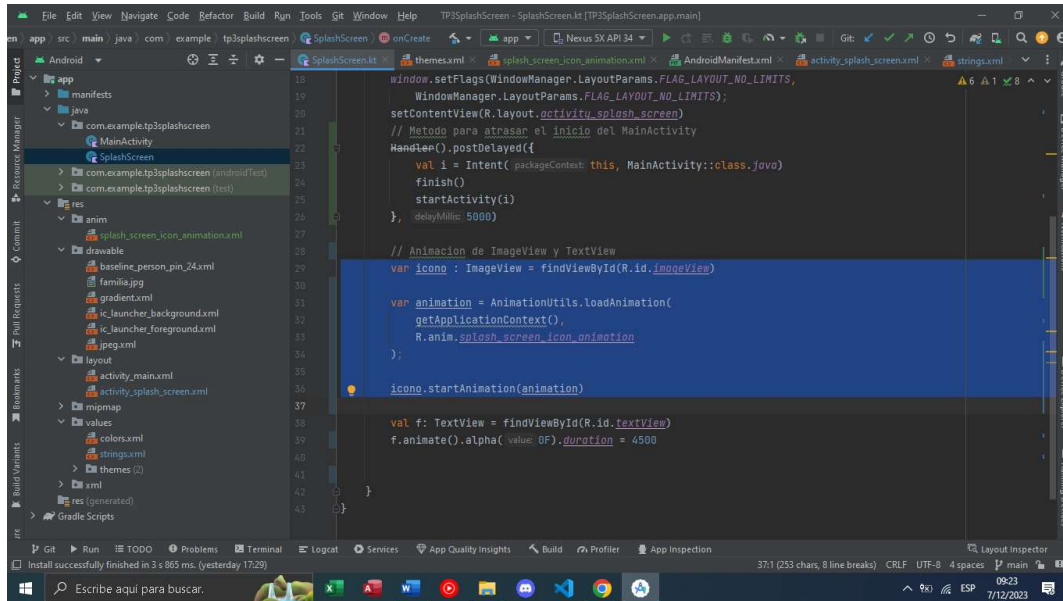
Hago un breve resumen de los pasos realizados para la animación. Primero que nada cree una carpeta dentro de res llamada anim, donde dentro cree un archivo XML llamado splash_screen_icon_animation.xml.

Este archivo se va a encargar de coordinar la animación del icono principal.



El ítem <Alpha> se encarga de hacer un apagado/encendido del elemento al que se le aplique. En este caso vamos de transparente (Alpha = 0) a totalmente visible (Alpha = 1) con una duración de 1500 milisegundos.

El ítem <rotate> indica una rotación del elemento. En este caso indicamos que va del ángulo 0 al ángulo 360 con una duración de 3000 ms con un retardo de 1500. El pivotX y pivotY indican el punto de rotación. 50% quiere decir que el punto de rotación va a estar a la mitad en X y a la mitad del objeto en Y, por lo tanto va a rotar sobre su centro geométrico. Para aplicar esta animación es menester avisarle al objeto que debe usar esta animación. Para ello volvemos al SplashScreen.kt y agregamos unas líneas de código mas:



```
18 window.setFlags(WindowManager.LayoutParams.FLAG_LAYOUT_NO_LIMITS,
19                 WindowManager.LayoutParams.FLAG_LAYOUT_NO_LIMITS);
20 setContentView(R.layout.activity_splash_screen)
21 // Metodo para atrasar el inicio del MainActivity
22 Handler().postDelayed({
23     val i = Intent( packageContext, this, MainActivity::class.java)
24     finish()
25     startActivity(i)
26 }, delayMillis: 5000)
27
28 // Animacion de ImageView y TextView
29 var icono : ImageView = findViewById(R.id.imageView)
30
31 var animation = AnimationUtils.loadAnimation(
32     getApplicationContext(),
33     R.anim.splash_screen_icon_animation
34 );
35
36 icono.startAnimation(animation)
37
38 val f: TextView = findViewById(R.id.textView)
39 f.animate().alpha( value: 0F, duration = 4500
40
41
42 }
43 }
```

Primero asigno a una variable el icono. Luego asigno la animación a otra variable mediante el método loadAnimation de la clase AnimationUtils donde debo pasarle el contexto y el recurso de animación que voy a asignarle.

Los views tienen un método que se llama startAnimation donde se le pasa como parámetro la animación la cual se quiere iniciar.

Las líneas 18 y 19 hacen de una forma mucho mas sencilla el apagado del texto del SplashScreen ya que es un fade sencillo.

Con todo esto logramos crear una pantalla de inicio personalizada con una duración de 5 segundos con 2 Views animados.

