



## **“Tarea individual”**

---

Francisco Burelo Murillo

Ingeniería En Sistemas Y Diseño De Software, Instituto Universitario de Yucatán

24040801 : Sistemas Operativos

Perla Alejandra Landero Heredia

10 de Agosto de 2025

---

```

1 public class A implements Runnable {
2     String palabra;
3
4     public A (String _palabra) {
5         palabra = _palabra; // Se inicializa la variable
6                               'palabra' con el argumento recibido
7     }
8
9     public void run () {
10        for (int i=0; i<100; i++) // Bucle que se repetirá 100
11                                   veces
12            System.out.println(palabra); // Imprime la palabra
13                                           asignada en cada iteración
14    }
15
16    public static void main (String args[]) {
17        A al = new A("al"); // Crea un objeto de la clase A con
18                              la palabra "al"

```

## Definicion de la clase que implementa Runnable

```
Public class A implements Runnable {
```

String palabra;

Se define la clase A, que implementa la interfaz Runnable , lo cual permite que objetos de esta clase puedan ser ejecutados como hilos.

Se declara una variable de instancia palabra, que almacenara el texto que se va a imprimir.

### Constructor de la clase

```
public A(String_palabra){
```

```
palabra=_palabra; //Se inicializa la variable 'palabra' con el argumento recibido
```

}

Este es el constructor de la clase A

Toma un parámetro\_palabra y lo asigna a la variable de instancia palabra-

## Método run()

```
public void run () {
```

```
for (int i=0;i<100;i++)//Bucle que se repetirá 100 veces
```

```
System.out.println(palabra); //Imprime la palabra asignada en cada iteración
```

Este es el método que se ejecutara cuando el hilo sea iniciado.

Hace un bucle de 100 iteraciones, imprimiendo la palabra almacenada en cada una

## Metodo main

```
Public static void main(String args[]){
```

```
A a1=newA("a"); // Crea un objeto de la clase A con la palabra "a1"
```

```
A a2=newA("a2"); // Crea otro objeto con la palabra "a2"
```

Se crean dos instancias de A, cada una con una palabra distinta "a1" y "a2"

## Creación de los hilos

```
Thread t1 = new Thread(a1); //Crea un hilo t1 usando el objeto a1
```

```
Thread t2 =new Thread(a2); //Crea otro hilo t2 usando el objeto a2
```

## Asignacion de prioridades

```
T1.setPriority(1); //Prioridad mínima
```

```
T2.setPriority(5); //Prioridad media
```

Se asignan prioridades a los hilos. En java:

1 es la prioridad mas baja

10 es la mas alta

5 es la prioridad por defecto

Impresión de las prioridades

```
System.out.println("Prioridad de t1:" +t1.getPriority());
```

```
System.out.println("Prioridad de t2:" +t2.getPriority());
```

Muestra en consola las prioridades que tienen los hilos t1 y t2.

Inicio de los hilos

```
T1.start();//Inicia el hilo t1 que imprimirá "a1"
```

```
T2.start();//Inicia el hilo t2 que imprimirá "a2"
```

```
}
```

Salida observada en consola

Prioridad de t1:1

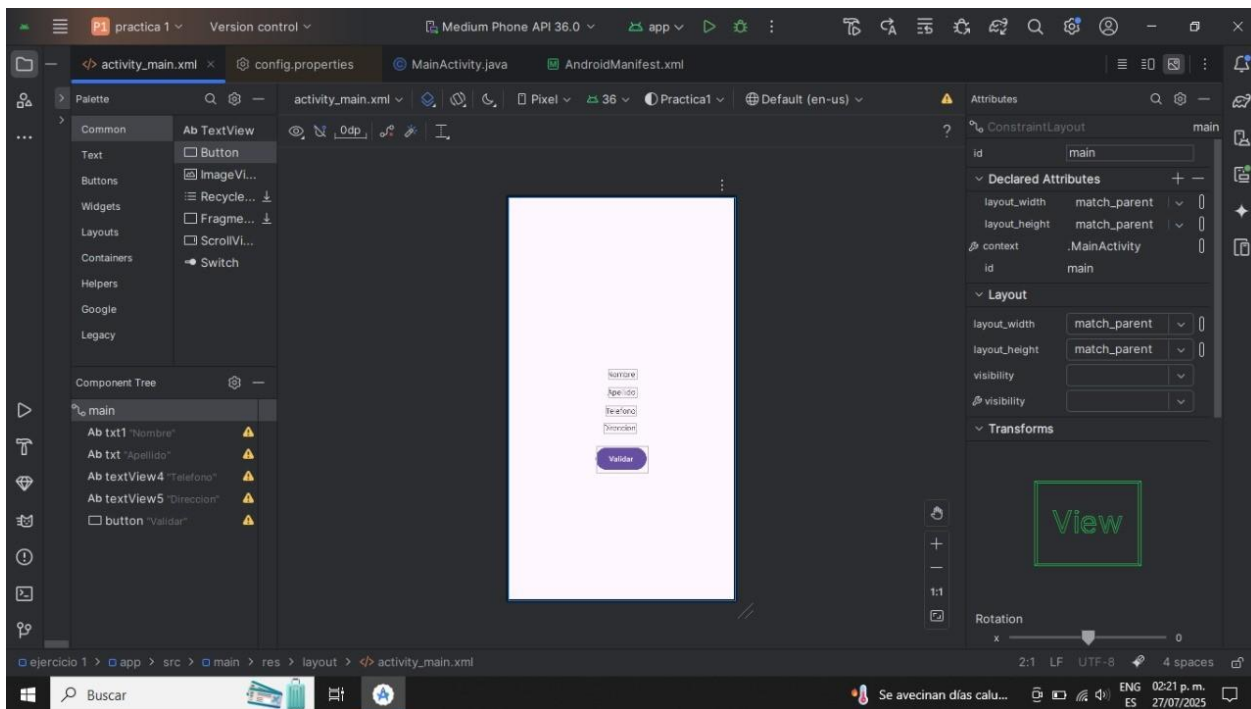
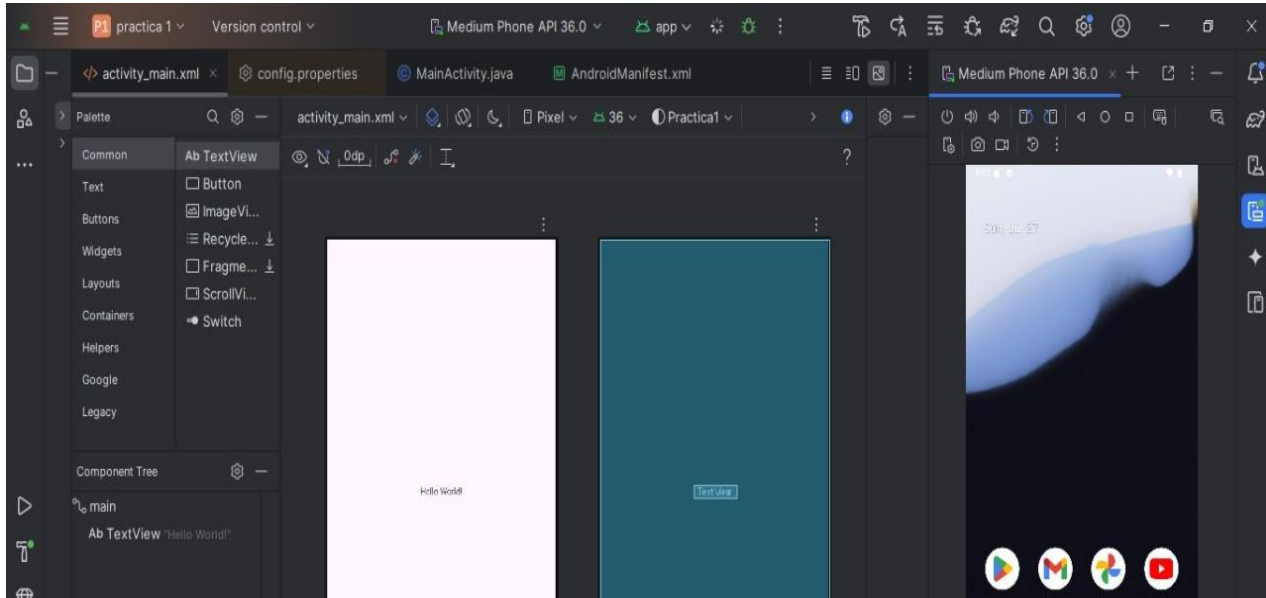
Prioridad de t2:5

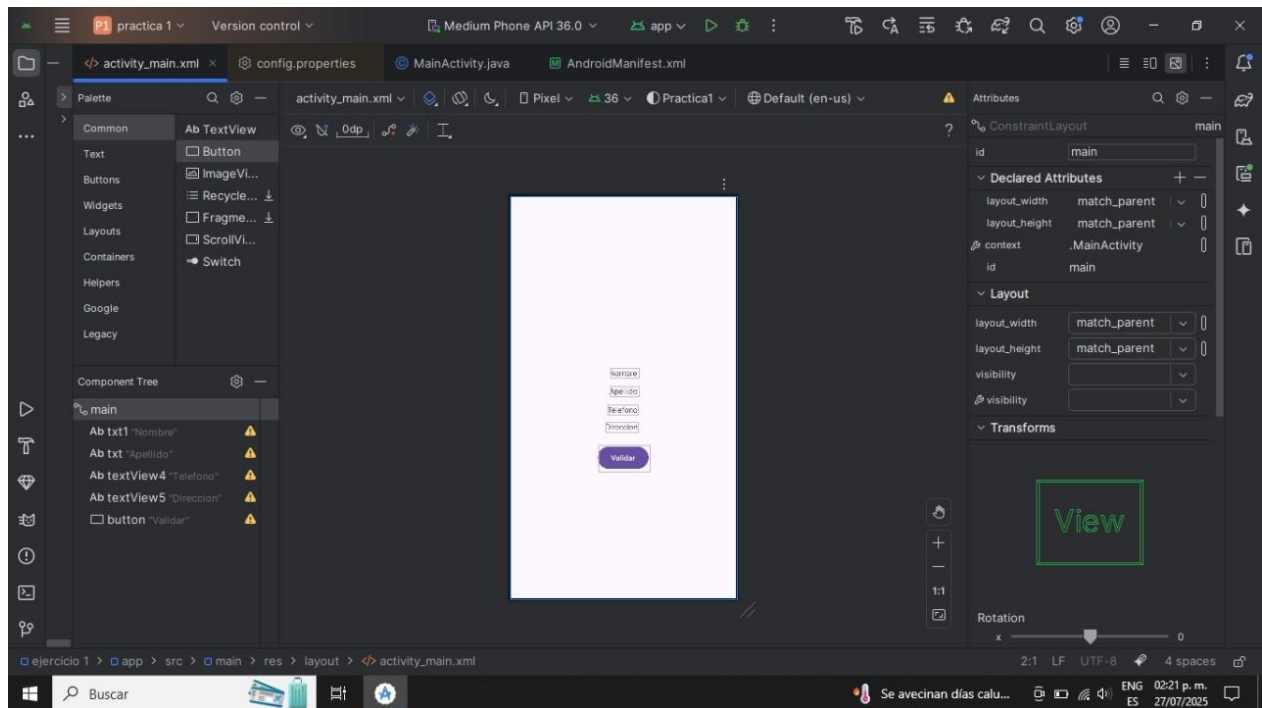
A2

A2

A2

# ANDROID Y JAVA





## CONCLUSION

En conclusión, este ejercicio proporciona una introducción clara, funcional y práctica a la programación concurrente en Java. Permite entender cómo se definen tareas paralelas, cómo se ejecutan mediante hilos, cómo influye (o no) la prioridad de los hilos en su comportamiento, y cómo gestionar la impresión concurrente en consola. A medida que se desarrollen programas más complejos, esta base será fundamental para abordar desafíos como la sincronización, la gestión de recursos compartidos y la prevención de condiciones de carrera. Por ello, este ejemplo no solo enseña cómo implementar hilos, sino que prepara al programador para pensar de manera paralela y estructurada, habilidades clave en el desarrollo de aplicaciones modernas y eficientes.

## **BIBLIOGRAFIA**

<https://www.programiz.com/java-programming/online-compiler/>