****

**“INDIVIDUAL”**



Perla Judith arias reyes

Licenciatura en ingeniería en sistema y

diseño de software,

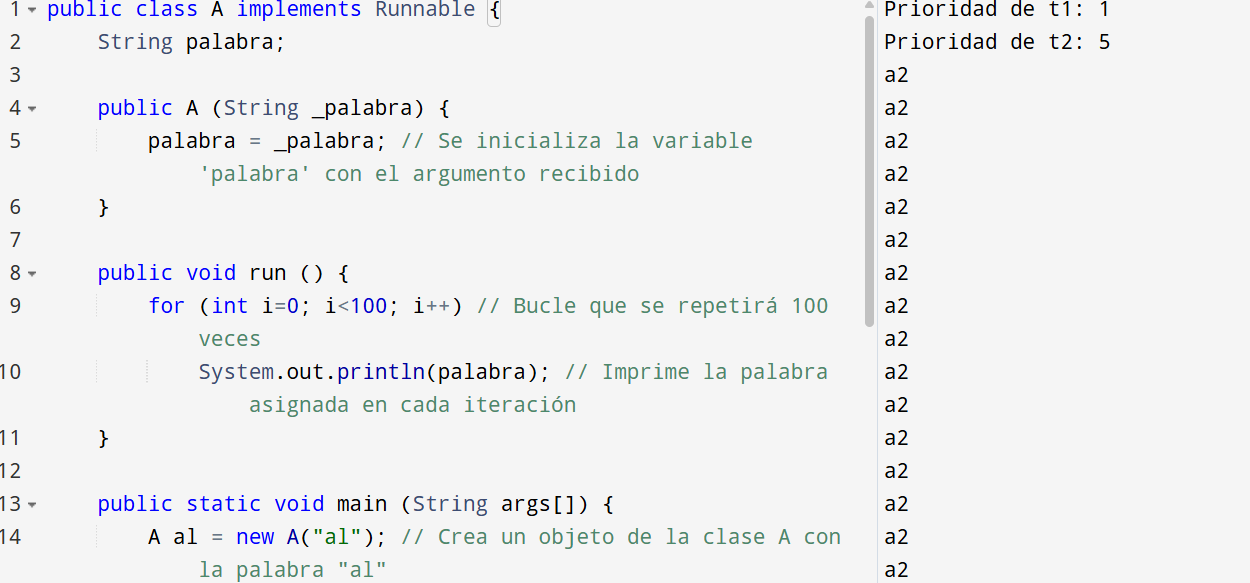
Instituto Universitario de Yucatán

24040741: SISTEMAS OPERATIVOS

Ing. PERLA ALEJANDRA LANDERO HEREDIA 09 DE JULIO 2025



Actividad 2. Realizar el siguiente código de manera individual y explicar que realiza en cada línea de sintaxis. Ingresar al siguiente enlace para realizar el ejercicio. <https://www.programiz.com/java-programming/online-compiler/>



**Definición 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 almacenará 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 ejecutará cuando el hilo sea iniciado.
* Hace un bucle de 100 iteraciones, imprimiendo la palabra almacenada en cada una.

**Método main**

public static void main (String args[]) {

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

A a2 = new A("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

**Asignación de prioridades**

t1.setPriority(1); // Prioridad mínima

t2.setPriority(5); // Prioridad media

* Se asignan prioridades a los hilos. En Java:
  + **1 es la prioridad más baja**
  + **10 es la más 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

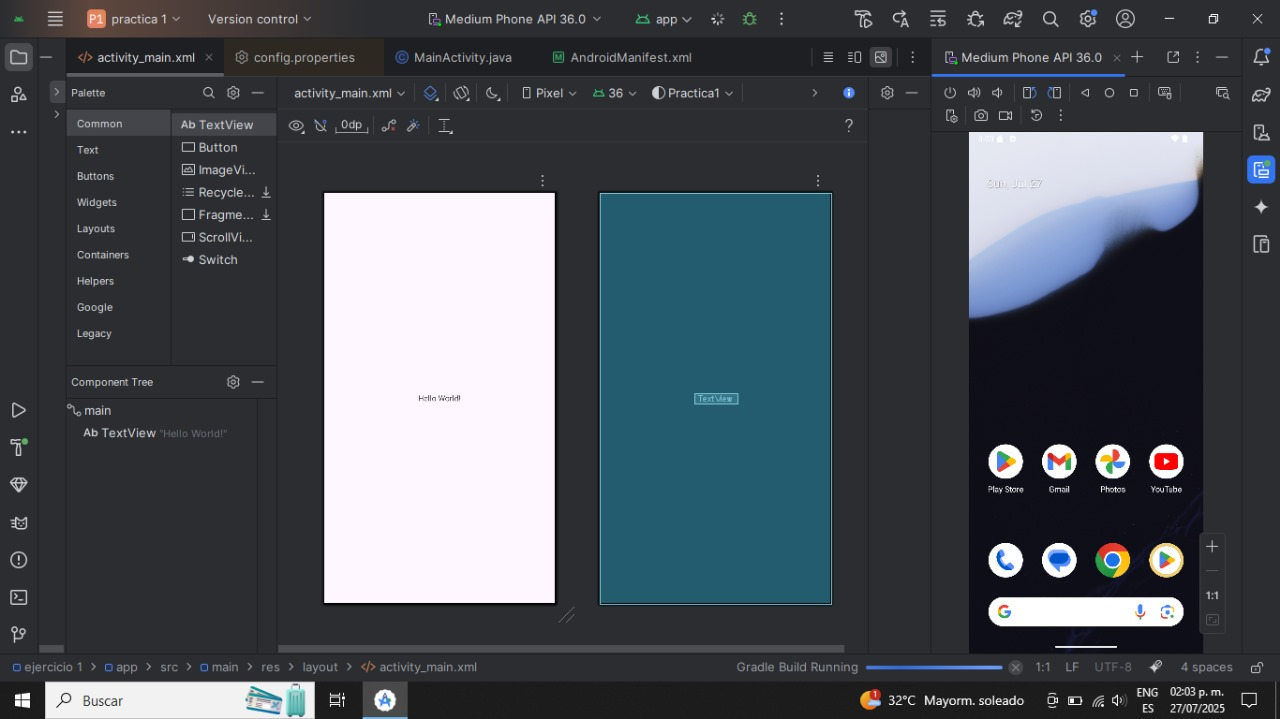
Prioridad de t2: 5

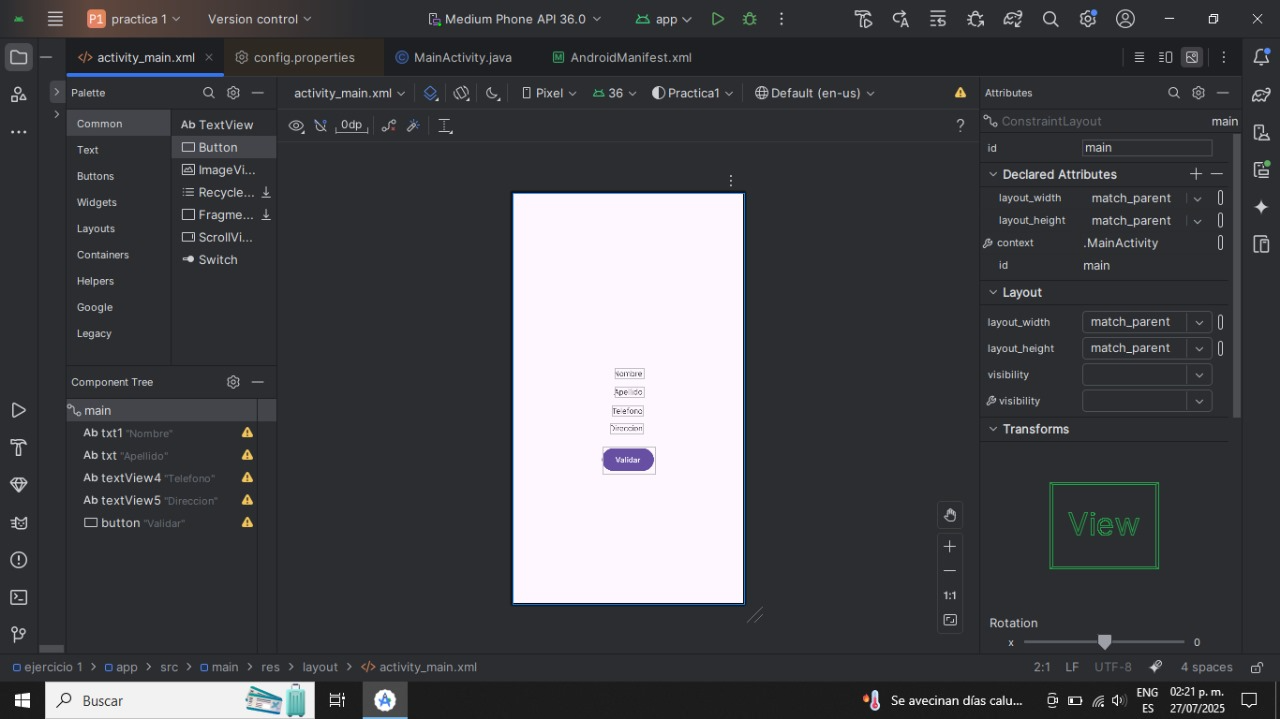
a2

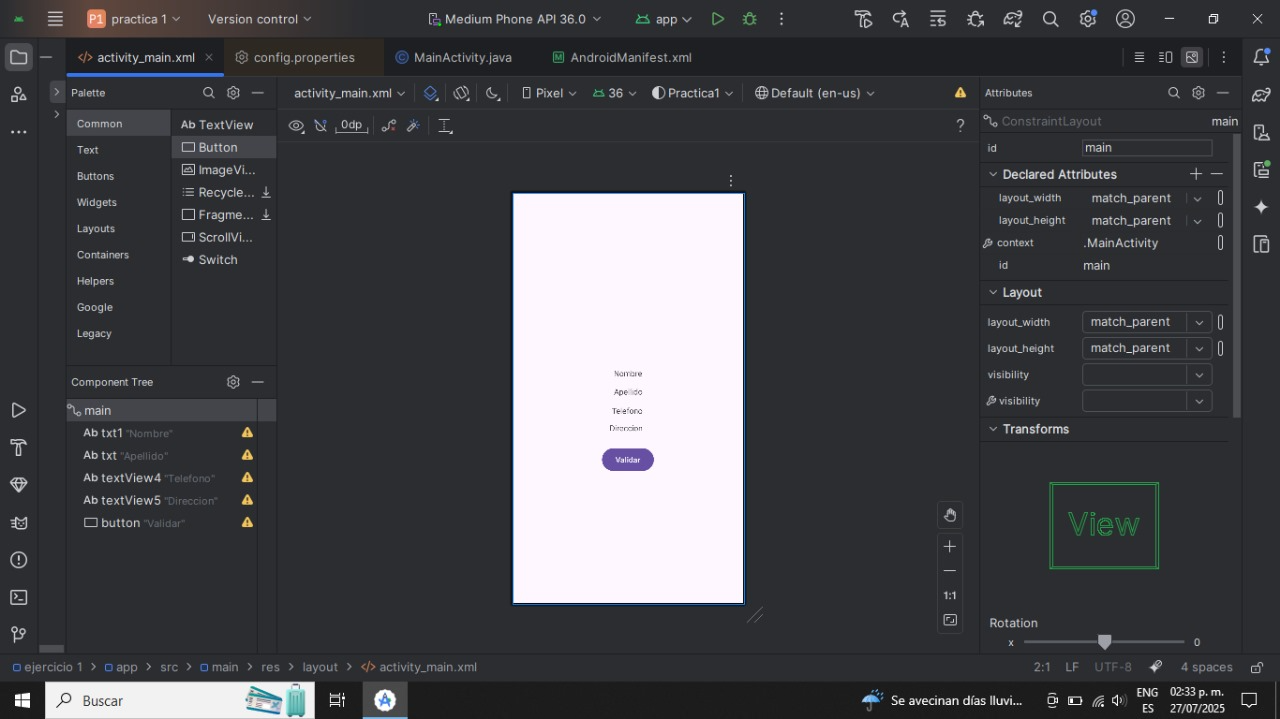
a2

a2

**Actividad 4.** De manera individual en la aplicación de Android Studio que instalaron realizar el siguiente ejercicio:







**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/>