@FunctionalInterface

public interface Runnable {

public abstract void run(); // Single abstract method

}

Runnable task = () -> System.out.println("Running in thread: " + Thread.currentThread().getName());

Thread t = new Thread(task);

t.start();

public class MyTask implements Runnable {

private final int taskId;

public MyTask(int taskId) {

this.taskId = taskId;

}

@Override

public void run() {

System.out.println("Task " + taskId + " is running on " + Thread.currentThread().getName());

try {

Thread.sleep(1000); // Simulate long task

} catch (InterruptedException e) {

Thread.currentThread().interrupt();

}

System.out.println("Task " + taskId + " completed.");

}

}

import java.util.concurrent.ExecutorService;

import java.util.concurrent.Executors;

public class ExecutorServiceMain {

public static void main(String[] args) {

ExecutorService executor = Executors.newFixedThreadPool(3);

for (int i = 1; i <= 5; i++) {

executor.submit(new MyTask(i)); // Submit separate task instance

}

executor.shutdown();

}

}

Task 1 is running on pool-1-thread-1

Task 2 is running on pool-1-thread-2

Task 3 is running on pool-1-thread-3

Task 1 completed.

Task 4 is running on pool-1-thread-1

Task 2 completed.

**✅ What is ScheduledExecutorService?**

It’s a subinterface of ExecutorService used for **scheduling tasks** to:

* Run **once after a delay**
* Run **periodically at fixed rate or fixed delay**

📦 Package: java.util.concurrent.ScheduledExecutorService

import java.time.LocalTime;

public class ScheduledTask implements Runnable {

private final String taskName;

public ScheduledTask(String taskName) {

this.taskName = taskName;

}

@Override

public void run() {

System.out.println("[" + LocalTime.now() + "] Executing task: " + taskName + " on " + Thread.currentThread().getName());

}

}

import java.util.concurrent.\*;

public class SchedulerMain {

public static void main(String[] args) {

ScheduledExecutorService scheduler = Executors.newScheduledThreadPool(2);

// Run once after 3 seconds

scheduler.schedule(new ScheduledTask("One-time task"), 3, TimeUnit.SECONDS);

// Run every 5 seconds after an initial delay of 2 seconds

scheduler.scheduleAtFixedRate(

new ScheduledTask("FixedRate task"),

2, // initial delay

5, // period between starts

TimeUnit.SECONDS

);

// Run every 7 seconds after finishing the previous task

scheduler.scheduleWithFixedDelay(

new ScheduledTask("FixedDelay task"),

1, // initial delay

7, // delay between end of one run and start of the next

TimeUnit.SECONDS

);

// Let it run for 20 seconds before shutting down

Executors.newSingleThreadScheduledExecutor().schedule(() -> {

System.out.println("Shutting down...");

scheduler.shutdown();

}, 20, TimeUnit.SECONDS);

}

}