# SingletonPatternExample:

Logger.java:

public class Logger {

// Create a private static instance

private static Logger instance;

// Private constructor to prevent instantiation

private Logger() {

System.out.println("Logger instance created");

}

// Public static method to provide access to the instance

public static Logger getInstance() {

if (instance == null) {

instance = new Logger();

}

return instance;

}

// Example logging method

public void log(String message) {

System.out.println("[LOG]: " + message);

}

}

TestLogger.java:

public class TestLogger {

public static void main(String[] args) {

// Attempt to get Logger instance multiple times

Logger logger1 = Logger.getInstance();

Logger logger2 = Logger.getInstance();

// Use the logger

logger1.log("This is the first log message.");

logger2.log("This is the second log message.");

// Check if both references point to the same object

if (logger1 == logger2) {

System.out.println("Only one Logger instance is used across the application.");

} else {

System.out.println("Multiple instances of Logger exist.");

}

}

}

