**Implementing the Singleton Pattern Example**

**Logger.java**

public class Logger {

private static Logger singleInstance;

private Logger() {

System.out.println("Logger Instance Created");

}

public static Logger getInstance() {

if (singleInstance == null) {

singleInstance = new Logger();

}

return singleInstance;

}

public void log(String message) {

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

}

}

**LoggerTest.java**

public class LoggerTest {

public static void main(String[] args) {

Logger logger1 = Logger.getInstance();

Logger logger2 = Logger.getInstance();

logger1.log("First log message");

logger2.log("Second log message");

if (logger1 == logger2) {

System.out.println("Both logger1 and logger2 refer to the same instance.");

}

}

}

**ThreadSafeLogger.java**

public class ThreadSafeLogger {

private static ThreadSafeLogger instance;

private ThreadSafeLogger() {

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

}

public static synchronized ThreadSafeLogger getInstance() {

if (instance == null) {

instance = new ThreadSafeLogger();

}

return instance;

}

public void log(String msg) {

System.out.println("[THREAD-SAFE LOG]: " + msg);

}

}

**ThreadSafeLoggerTest.java**

public class ThreadSafeLoggerTest {

public static void main(String[] args) {

ThreadSafeLogger logger1 = ThreadSafeLogger.getInstance();

ThreadSafeLogger logger2 = ThreadSafeLogger.getInstance();

logger1.log("First thread-safe log");

logger2.log("Second thread-safe log");

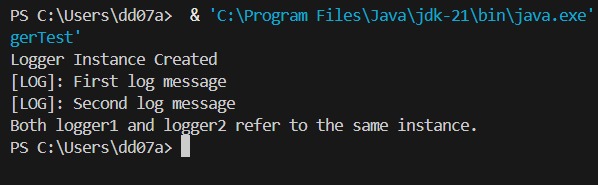
if (logger1 == logger2) {

System.out.println("Both instances are the same (Thread-Safe Singleton).");

}

}

}

**Output:  
**