Code: Logger.cs

using System;

public class Logger

{

// Private static instance of Logger

private static Logger? instance = null;

// Object for locking to make thread-safe

private static readonly object padlock = new object();

// Private constructor to prevent external instantiation

private Logger()

{

Console.WriteLine("Logger instance created.");

}

// Public static method to get the singleton instance

public static Logger GetInstance()

{

if (instance == null)

{

lock (padlock)

{

if (instance == null)

{

instance = new Logger();

}

}

}

return instance;

}

// Example logging method

public void Log(string message)

{

Console.WriteLine($"[LOG] {message}");

}

}

Program.cs:-

using System;

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Testing Singleton Logger:");

Logger logger1 = Logger.GetInstance();

Logger logger2 = Logger.GetInstance();

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

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

// Check if both references point to the same instance

if (logger1 == logger2)

{

Console.WriteLine("Both logger1 and logger2 are the same instance.");

}

else

{

Console.WriteLine("Different instances detected (not a singleton).");

}

}

}

Output:-

