Exercise 1: Implementing the Singleton Pattern

Code:

using System;

namespace SingletonPatternExample

{

// Singleton Class

public class Singleton

{

// Lazy<T> handles thread safety and lazy initialization

private static readonly Lazy<Singleton> \_lazyInstance = new Lazy<Singleton>(() => new Singleton());

// Private constructor to prevent external instantiation

private Singleton()

{

Console.WriteLine("Singleton Instance Created");

}

// Public property to access the single instance

public static Singleton Instance

{

get

{

return \_lazyInstance.Value;

}

}

// A sample method to demonstrate functionality

public void ShowMessage()

{

Console.WriteLine("Hello from Singleton!");

}

}

// Test Program

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Accessing Singleton Instance...");

Singleton firstInstance = Singleton.Instance;

firstInstance.ShowMessage();

Singleton secondInstance = Singleton.Instance;

secondInstance.ShowMessage();

// Check if both references point to the same instance

Console.WriteLine("Are both instances equal? " + (firstInstance == secondInstance)); // Output: True

Console.ReadLine(); // Keeps console open

}

}

}

