using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp8

{

Sealed class Singleton

{

private static int COunt = 0;

private static Singleton instance = null;

public static Singleton getInstance

{

get

{

if (instance == null)

instance = new Singleton();

return instance;

}

}

private Singleton()

{

COunt++;

Console.WriteLine("Valus o counteris " + COunt);

}

public void PrintDEtails(string message)

{

Console.WriteLine(message);

}

}

class Program

{

static void Main(string[] args)

{

Singleton singleton = Singleton.getInstance;

singleton.PrintDEtails("rmpl");

Singleton singleton1 = Singleton.getInstance;

singleton1.PrintDEtails("st");

}

}

}

Thread Safety

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp8

{

class Singleton

{

private static int COunt = 0;

private static readonly Object obj = new object();

private static Singleton instance = null;

public static Singleton getInstance

{

get

{

if (instance == null)

{

lock (obj)

{

if (instance == null)

instance = new Singleton();

}

}

return instance;

}

}

private Singleton()

{

COunt++;

Console.WriteLine("Valus o counteris " + COunt);

}

public void PrintDEtails(string message)

{

Console.WriteLine(message);

}

}

class Program

{

static void Main(string[] args)

{

Parallel.Invoke(() => PrintEmployees(), ()=> PrintStudents());

//PrintEmployees();

//PrintStudents();

}

private static void PrintStudents()

{

Singleton singleton1 = Singleton.getInstance;

singleton1.PrintDEtails("st");

}

private static void PrintEmployees()

{

Singleton singleton = Singleton.getInstance;

singleton.PrintDEtails("rmpl");

}

}

}