using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

namespace SignalMechanism

{

public class Program

{

//static void TaskMethod(string name)

//{

// Console.WriteLine($@"TaskMethod {name} is running id {Thread.CurrentThread.ManagedThreadId}

// Is Threadpool Thread {Thread.CurrentThread.IsThreadPoolThread}");

//}

//private static void Main(string[]args)

//{

// Console.WriteLine($@"Main is running {Thread.CurrentThread.ManagedThreadId}

// Is Threadpool Thread {Thread.CurrentThread.IsThreadPoolThread}");

// Task t1 = new Task(() =>

// {

// TaskMethod("Task 1");

// });

// t1.Start();

// Task t2 = new Task(() =>

// {

// TaskMethod("Task 2");

// });

// t2.Start();

// //OYREN

// //TaskCreationOptions.LongRunning

// // TaskCreationOptions.PreferFairness

// // TaskCreationOptions.AttachedToParent

// var t4 = Task.Factory.StartNew(() => { TaskMethod("Task 3"); });

// var t5 = Task.Factory.StartNew(() => { TaskMethod("Task 4 "); },TaskCreationOptions.LongRunning);

// Console.ReadLine();

//}

//static int GetSum(int end)

//{

// int sum = 0;

// for (int i = 0; i < end; i++)

// {

// sum += i;

// Thread.Sleep(1);

// Console.WriteLine("CALCULATION");

// }

// return sum;

//}

//static async void Start()

//{

// var task = new Task<int>(() =>

// {

// return GetSum(500);

// });

// task.Start();

// var result =await task;

// Console.WriteLine(result);

//}

//static void Main(string[] args)

//{

// Start();

// Console.ReadLine();

//}

#region Example 2 with Task

static Task<int> MyTaskMethodAsync()

{

Console.WriteLine($@"Method is running id : {Thread.CurrentThread.ManagedThreadId}");

var task = new Task<int>(() => TaskMethod("John", 10));

task.Start();

return task;

}

private static int TaskMethod(string v1, int v2)

{

Console.WriteLine($@"Method is running id : {Thread.CurrentThread.ManagedThreadId}");

return 42 \* v2;

}

static async Task Start()

{

Console.WriteLine($@"Main is running id : {Thread.CurrentThread.ManagedThreadId}");

var task = MyTaskMethodAsync();

//Console.WriteLine(task.Result);

Console.WriteLine(await task);

}

static async void Run()

{

await Start();

}

static void Main(string[] args)

{

Run();

Console.ReadLine();

}

#endregion

}

}