using System;

using System.Net.Http;

using System.Threading.Tasks;

namespace AsyncApp

{

class Program

{

private const string URL = "https://www.uakron.edu";

static void Main(string[] args)

{

DoSynchronousWork();

var someTask = DoSomethingAsync();

DoSynchronousWorkAfterAwait();

someTask.Wait(); // this is blocking call, used it only on Main method

Console.ReadLine();

}

public static void DoSynchronousWork()

{

// You can do whatever work is needed here

Console.WriteLine("1. Doing some work synchronously");

}

static async Task DoSomethingAsync() // A task return type will ventually yield a void

{

Console.WriteLine("2. Async task has started...");

await GetStringAsync();

}

static async Task GetStringAsync()

{

using (var httpClient = new HttpClient())

{

Console.WriteLine("3.Awaiting the result of GetStringAsync of Http Client...");

string result = await httpClient.GetStringAsync(URL);// execution pause here while awaiting GetSteing Async to complete

// From this line and below the execurom will resume onnce the above awaitable os done

// using await keyword ,it will do magic of unwraping teh Task,<string> into string (result variable)

Console.WriteLine();

Console.WriteLine("4. The awaited task has completed .Let's get the content length...");

Console.WriteLine($"5. The length of http Get for {URL}");

Console.WriteLine($"6. {result.Length} Total Number of Characters");

}

}

static void DoSynchronousWorkAfterAwait()

{

// This os the work we can do while waiting for the awaited Async Task to complete

Console.WriteLine("7. While waiting for the async task to finish ,we can do some unrelated work...");

Console.WriteLine();

for (var i = 0; i <= 5; i++)

{

for (var j = i; j <= 5; j++)

{

Console.Write("GO Cavs");

}

Console.WriteLine();

}

}

}

}