**Hands-On: Stage 3 - C# Additional Topics Async Programming, Multithreading - Day 3 – Handson 1**

**Async Await usage – 1**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

namespace AsyncHandsOn

{

    class Program

    {

        public static async Task firstMethod()

        {

            await Task.Run(() =>

            {

                string a = secondMethod();

                Console.WriteLine(a);

            });

        }

        public static string secondMethod()

        {

            Console.WriteLine("wait for string to return");

            Thread.Sleep(4444);

            return "this is the second method ";

        }

        static void Main(string[] args)

        {

            firstMethod();

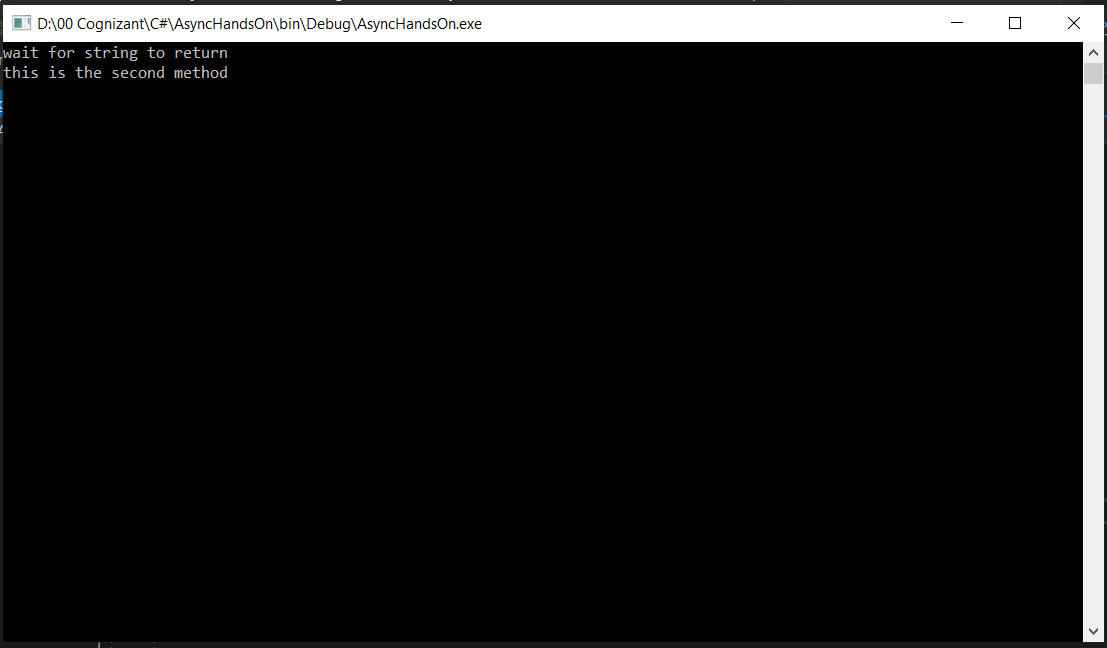
            Console.ReadKey();

        }

    }

}

**OUTPUT**



**Async Await usage – 2**

**Form1.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp

{

    public partial class Form1 : Form

    {

        public Form1()

        {

            InitializeComponent();

        }

        public int CountChars()

        {

            int count = 0;

            using (StreamReader streamReader = new StreamReader("D:\\Day3\_Handson\\Kuttralam.txt"))

            {

                string content = streamReader.ReadToEnd();

                count = content.Length;

                Thread.Sleep(2000);

            }

            return count;

        }

        private void label1\_Click(object sender, EventArgs e)

        {

        }

        private void Form1\_Load(object sender, EventArgs e)

        {

        }

        private async void button1\_Click\_1(object sender, EventArgs e)

        {

            Task<int> task = new Task<int>(CountChars);

            task.Start();

            label1.Text = "File is processing";

            int count = await task;

            label1.Text = count.ToString() + " characters";

        }

    }

}

**Form1.Designer.cs**

namespace WindowsFormsApp

{

    partial class Form1

    {

        /// <summary>

        ///  Required designer variable.

        /// </summary>

        private System.ComponentModel.IContainer components = null;

        /// <summary>

        ///  Clean up any resources being used.

        /// </summary>

        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

        protected override void Dispose(bool disposing)

        {

            if (disposing && (components != null))

            {

                components.Dispose();

            }

            base.Dispose(disposing);

        }

        #region Windows Form Designer generated code

        /// <summary>

        ///  Required method for Designer support - do not modify

        ///  the contents of this method with the code editor.

        /// </summary>

        private void InitializeComponent()

        {

            this.label1 = new System.Windows.Forms.Label();

            this.button1 = new System.Windows.Forms.Button();

            this.SuspendLayout();

            //

            // label1

            //

            this.label1.AutoSize = true;

            this.label1.Location = new System.Drawing.Point(300, 150);

            this.label1.Name = "label1";

            this.label1.Size = new System.Drawing.Size(0, 20);

            this.label1.TabIndex = 1;

            //

            // button1

            //

            this.button1.Location = new System.Drawing.Point(643, 381);

            this.button1.Name = "button1";

            this.button1.Size = new System.Drawing.Size(94, 29);

            this.button1.TabIndex = 2;

            this.button1.Text = "button1";

            this.button1.UseVisualStyleBackColor = true;

            this.button1.Click += new System.EventHandler(this.button1\_Click\_1);

            //

            // Form1

            //

            this.AutoScaleDimensions = new System.Drawing.SizeF(8F, 20F);

            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

            this.ClientSize = new System.Drawing.Size(800, 450);

            this.Controls.Add(this.button1);

            this.Controls.Add(this.label1);

            this.Name = "Form1";

            this.SizeGripStyle = System.Windows.Forms.SizeGripStyle.Show;

            this.Text = "Form1";

            this.Load += new System.EventHandler(this.Form1\_Load);

            this.ResumeLayout(false);

            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.Label label1;

        private System.Windows.Forms.Button button1;

    }

}

**Program.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp

{

    static class Program

    {

        [STAThread]

        static void Main()

        {

            Application.EnableVisualStyles();

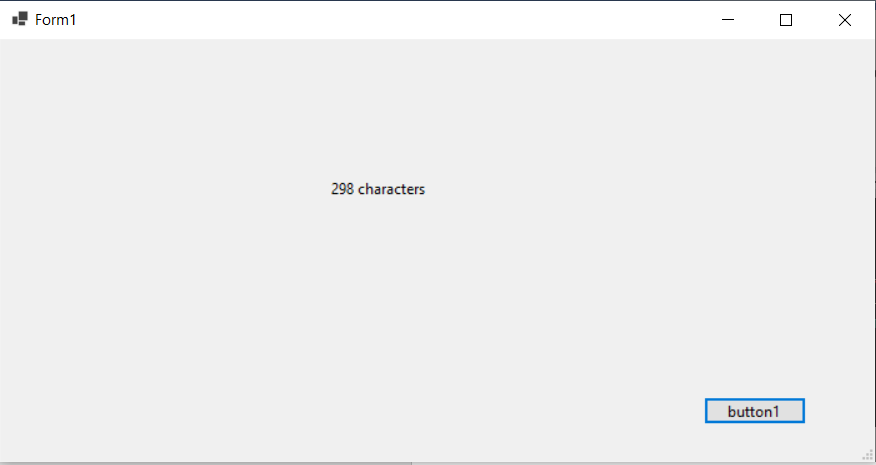
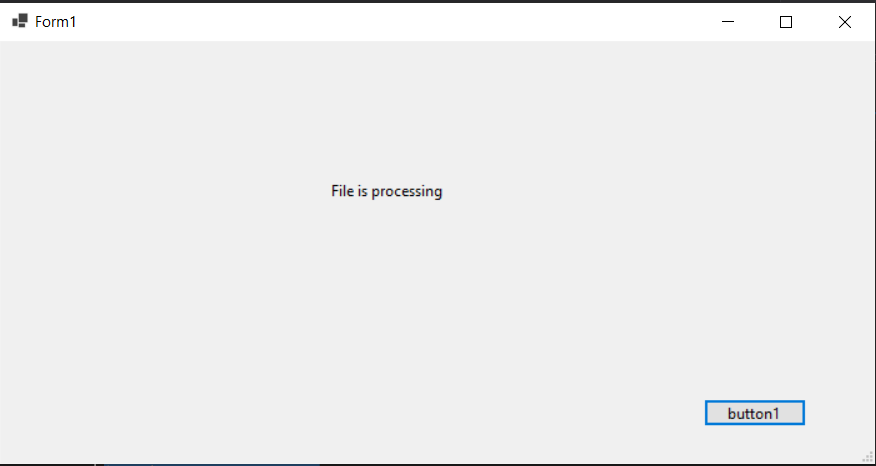
            Application.SetCompatibleTextRenderingDefault(false);

            Application.Run(new Form1());

        }

    }

}

**OUTPUT**

**Kuttralam.txt**

A paragraph is a series of related sentences developing a central idea, called the topic. Try to think about paragraphs in terms of thematic unity: a paragraph is a sentence or a group of sentences that supports one central, unified idea. Paragraphs add one idea at a time to your broader argument.

**Named parameters – Order of arguments as per the function and modify**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Handson4.\_5

{

    class NamedParameter

    {

        static void GetCohortDetails(string Cohortname, int Genccount, string Mode, string Track, string CurrentModule)

        {

            Console.WriteLine("It is {0} with {1} GenCs undergoing training for {2} thru {3}. The current module of training is {4}", Cohortname, Genccount, Track, Mode, CurrentModule);

        }

        // Optional Prameters

        static void OrderDetails(string Productname, string Sellername, int Orderquantity = 1, bool returnable = true)

        {

            Console.WriteLine("Here is the order detail – {0} number of {1} by {2} is ordered. It’s returnable status is {3}", Orderquantity, Productname, Sellername, returnable);

        }

        static void Main(string[] args)

        {

            Console.WriteLine("Cohort Details");

            // Named parameters

            GetCohortDetails(Cohortname: "CDE", Genccount: 18, Track: "Java", Mode: "OBL", CurrentModule: "Stage 3");

            GetCohortDetails(Cohortname: "CDE", Genccount: 18, Mode: "PARC", Track: ".Net", CurrentModule: "Stage 3");

            Console.WriteLine("-------------------------------------");

            // Optional parameters

            Console.WriteLine("Order  Details");

            OrderDetails(Sellername: "abc", Productname: "def", Orderquantity: 10, returnable: false);

            OrderDetails(Sellername: "abc", Productname: "def");

            Console.WriteLine("-------------------------------------");

            Console.ReadLine();

        }

    }

}

**OUTPUT**

