**Async\_Await Handson Solutions**

**Date:**18-06-2021

**Name:**Asritha Cherukuri

**EmpId:** 916179

**Handson**

Asyn Await usage-1

*Implementation:*

using System;

using System.Threading;

using System.Threading.Tasks;

namespace AsyncAwait

{

class Program

{

//Async Await usage - 1

public static async Task Method1()

{

await Task.Run(() =>

{

string s = Method2();

Console.WriteLine(s);

});

}

public static string Method2()

{

Console.WriteLine("The process is waiting to return the string");

Thread.Sleep(5000);

return "Hello from method2";

}

static void Main(string[] args)

{

Method1();

Console.ReadKey();

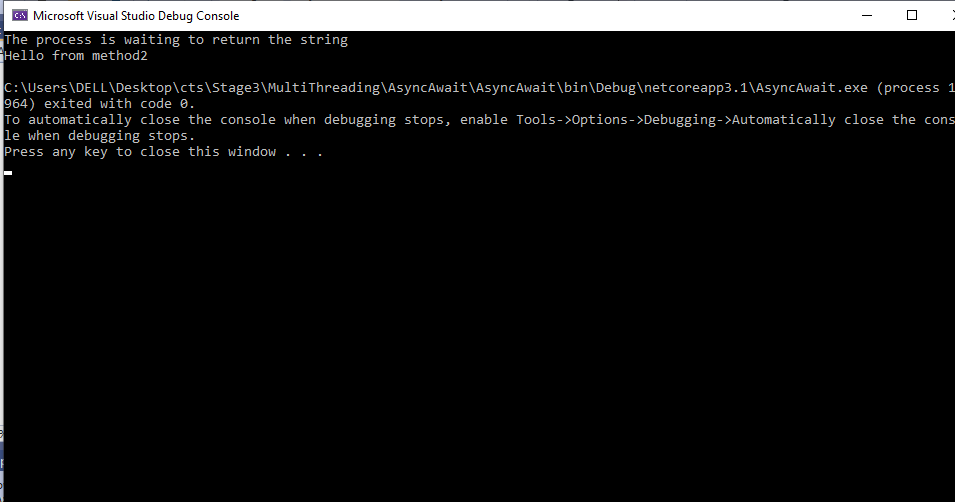
//Console.WriteLine("Hello World!");

}

}

}

*Output:*

**

Asyn Await usage-2

*Implementation:*

*Program.cs*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WinFormsApp1

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.SetHighDpiMode(HighDpiMode.SystemAware);

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

*Form1.Designer.cs*

namespace WinFormsApp1

{

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

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

this.SuspendLayout();

//

// button1

//

this.button1.AccessibleName = "button";

this.button1.Location = new System.Drawing.Point(553, 36);

this.button1.Name = "button1";

this.button1.Size = new System.Drawing.Size(75, 23);

this.button1.TabIndex = 0;

this.button1.Text = "Process";

this.button1.UseVisualStyleBackColor = true;

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

//

// label1

//

this.label1.AccessibleName = "label";

this.label1.AutoSize = true;

this.label1.Font = new System.Drawing.Font("Segoe UI", 10F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point);

this.label1.Location = new System.Drawing.Point(201, 106);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(0, 19);

this.label1.TabIndex = 1;

this.label1.Click += new System.EventHandler(this.label1\_Click);

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(7F, 15F);

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

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

this.Controls.Add(this.label1);

this.Controls.Add(this.button1);

this.Name = "Form1";

this.Text = "Form1";

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

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Button button1;

private System.Windows.Forms.Label label1;

}

}

*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 WinFormsApp1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private int CountCharacters()

{

int count = 0;

// Create a StreamReader and point it to the file to read

using (StreamReader reader = new StreamReader("C:\\Users\\DELL\\Desktop\\cts\\Stage3\\WindowsForms\\Data.txt"))

{

string content = reader.ReadToEnd();

count = content.Length;

// Make the program look busy for 5 seconds

Thread.Sleep(5000);

}

return count;

}

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

{

// Create a task to execute CountCharacters() function

// CountCharacters() function returns int, so we created Task<int>

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

task.Start();

label1.Text = "Processing file. Please wait...";

// Wait until the long running task completes

int count = await task;

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

}

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

{

}

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

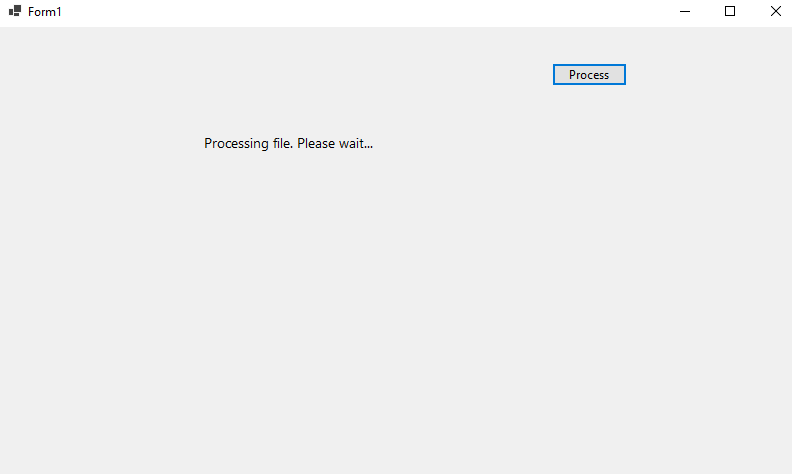
{

}

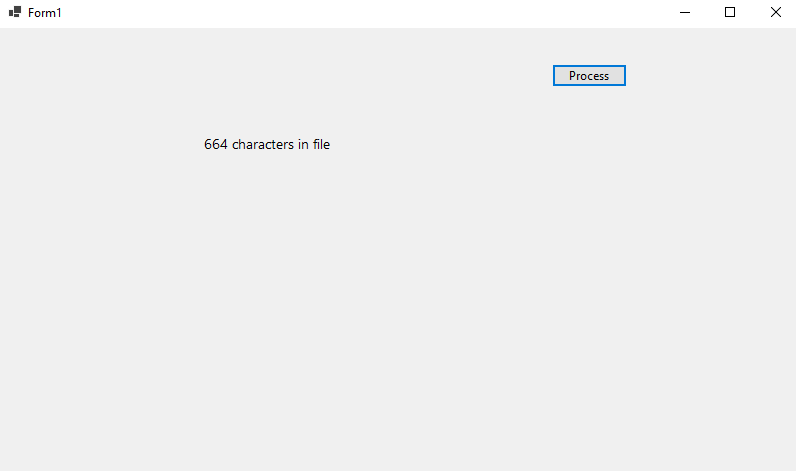
}

}

*Output:SS-1*

**

*SS-2*

**

*Data.txt*

*Once, there was a boy who became bored when he watched over the village sheep grazing on the hillside. To entertain himself, he sang out, “Wolf! Wolf! The wolf is chasing the sheep!”When the villagers heard the cry, they came running up the hill to drive the wolf away. But, when they arrived, they saw no wolf. The boy was amused when seeing their angry faces.“Don’t scream wolf, boy,” warned the villagers, “when there is no wolf!” They angrily went back down the hill.Later, the shepherd boy cried out once again, “Wolf! Wolf! The wolf is chasing the sheep!” To his amusement, he looked on as the villagers came running up the hill to scare the wolf away*

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

*Implementation:*

using System;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp2

{

class program

{

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

{

Console.WriteLine("COHORTNAME {0} NumberOfGenCs {1} Training- {2} Mode- {3} ModuleNumber {4}", Cohortname, Genccount, Track, Mode, CurrentModule);

}

// Optional Prameters

static void OrderDetails(string Product, string Seller, int Orderquantity = 1, bool returnable = true)

{

Console.WriteLine("order detail Orderquality– {0} product- {1} seller- {2} . Returnable status- {3}", Orderquantity, Product, Seller, 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("");

Console.WriteLine("");

// Optional parameters

Console.WriteLine("Order Details");

OrderDetails(Seller: "abc", Product: "def", Orderquantity: 10, returnable: false);

OrderDetails(Seller: "abc", Product: "def");

Console.WriteLine("");

Console.WriteLine("");

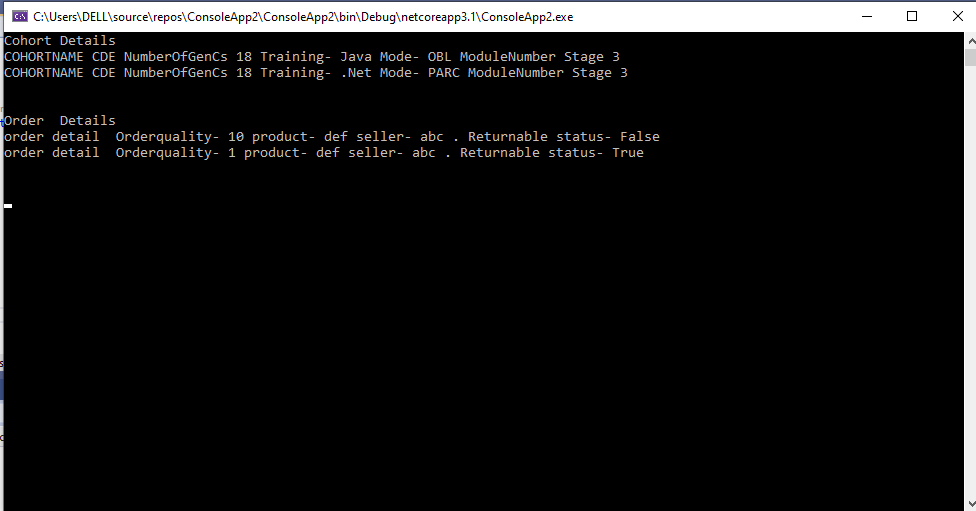
Console.ReadLine();

}

}

}

*Output:*

**