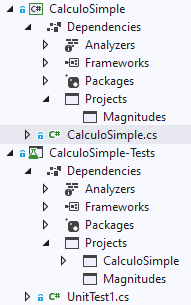
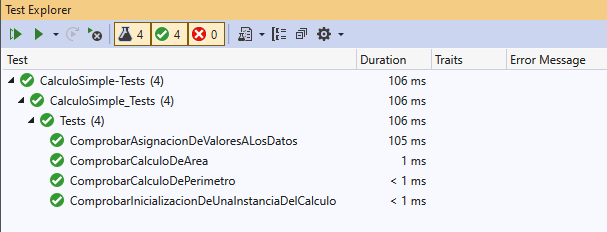
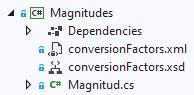
# pruebas y experimentos de “multi-interfaz” de usuario

creado 20210331T1216 guardado 20210331T1505 impreso 20210331T1505

## El programa base, que se va a utilizar desde diversos interfaces de usuario…

using System;

namespace CalculoSimple

{

public class CalculoSimple

{

private Magnitudes.Magnitud unlado;

private Magnitudes.Magnitud otrolado;

private Magnitudes.Magnitud perimetro;

private Magnitudes.Magnitud area;

public CalculoSimple()

{

unlado = new Magnitudes.Magnitud();

otrolado = new Magnitudes.Magnitud();

perimetro = new Magnitudes.Magnitud();

area = new Magnitudes.Magnitud();

}

private void ReCalcular()

{

if (!double.IsNaN(unlado.valor) && !double.IsNaN(otrolado.valor))

{

perimetro.valor = 2.0 \* unlado.valor + 2.0 \* otrolado.getCopiaConvertidaA(unlado.unidaddemedida).valor;

perimetro.unidaddemedida = unlado.unidaddemedida;

area.valor = unlado.valor \* otrolado.getCopiaConvertidaA(unlado.unidaddemedida).valor;

area.unidaddemedida = unlado.unidaddemedida + "2";

}

}

public void setDato\_unlado(Magnitudes.Magnitud dato)

{

unlado = dato;

ReCalcular();

}

public void setDato\_otrolado(Magnitudes.Magnitud dato)

{

otrolado = dato;

ReCalcular();

}

public Magnitudes.Magnitud getResultado\_perimetro()

{

return perimetro;

}

public Magnitudes.Magnitud getResultado\_area()

{

return area;

}

public Magnitudes.Magnitud getDato\_unlado()

{

return unlado;

}

public Magnitudes.Magnitud getDato\_otrolado()

{

return otrolado;

}

}

}

using NUnit.Framework;

namespace CalculoSimple\_Tests

{

public class Tests

{

[SetUp]

public void Setup()

{

}

[Test]

public void ComprobarInicializacionDeUnaInstanciaDelCalculo()

{

CalculoSimple.CalculoSimple calculadora = new CalculoSimple.CalculoSimple();

Assert.IsTrue(double.IsNaN(calculadora.getDato\_unlado().valor));

Assert.AreEqual(Magnitudes.Magnitud.DESCONOCIDA, calculadora.getDato\_unlado().unidaddemedida);

Assert.IsTrue(double.IsNaN(calculadora.getDato\_otrolado().valor));

Assert.AreEqual(Magnitudes.Magnitud.DESCONOCIDA, calculadora.getDato\_otrolado().unidaddemedida);

}

[Test]

public void ComprobarAsignacionDeValoresALosDatos()

{

CalculoSimple.CalculoSimple calculadora = new CalculoSimple.CalculoSimple();

calculadora.setDato\_unlado(new Magnitudes.Magnitud(valor: 2, unidadDeMedida: "m"));

Assert.AreEqual(2.0, calculadora.getDato\_unlado().valor);

Assert.AreEqual("m", calculadora.getDato\_unlado().unidaddemedida);

calculadora.setDato\_otrolado(new Magnitudes.Magnitud(valor: 300, unidadDeMedida: "cm"));

Assert.AreEqual(300.0, calculadora.getDato\_otrolado().valor);

Assert.AreEqual("cm", calculadora.getDato\_otrolado().unidaddemedida);

}

[Test]

public void ComprobarCalculoDePerimetro()

{

CalculoSimple.CalculoSimple calculadora = new CalculoSimple.CalculoSimple();

Assert.IsTrue(double.IsNaN(calculadora.getResultado\_perimetro().valor));

Assert.AreEqual(Magnitudes.Magnitud.DESCONOCIDA, calculadora.getResultado\_perimetro().unidaddemedida);

calculadora.setDato\_unlado(new Magnitudes.Magnitud(valor: 2, unidadDeMedida: "m"));

Assert.IsTrue(double.IsNaN(calculadora.getResultado\_perimetro().valor));

Assert.AreEqual(Magnitudes.Magnitud.DESCONOCIDA, calculadora.getResultado\_perimetro().unidaddemedida);

calculadora.setDato\_otrolado(new Magnitudes.Magnitud(valor: 300, unidadDeMedida: "cm"));

Assert.AreEqual(10, calculadora.getResultado\_perimetro().valor);

Assert.AreEqual("m", calculadora.getResultado\_perimetro().unidaddemedida);

}

[Test]

public void ComprobarCalculoDeArea()

{

CalculoSimple.CalculoSimple calculadora = new CalculoSimple.CalculoSimple();

Assert.IsTrue(double.IsNaN(calculadora.getResultado\_area().valor));

Assert.AreEqual(Magnitudes.Magnitud.DESCONOCIDA, calculadora.getResultado\_area().unidaddemedida);

calculadora.setDato\_unlado(new Magnitudes.Magnitud(valor: 2, unidadDeMedida: "m"));

Assert.IsTrue(double.IsNaN(calculadora.getResultado\_area().valor));

Assert.AreEqual(Magnitudes.Magnitud.DESCONOCIDA, calculadora.getResultado\_area().unidaddemedida);

calculadora.setDato\_otrolado(new Magnitudes.Magnitud(valor: 300, unidadDeMedida: "cm"));

Assert.AreEqual(6, calculadora.getResultado\_area().valor);

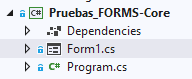
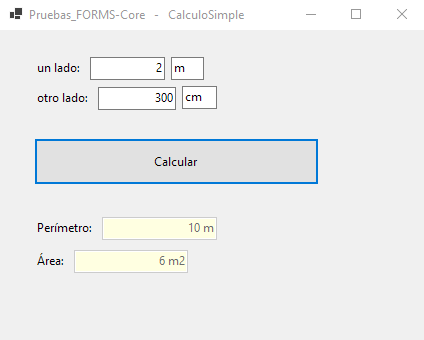
Assert.AreEqual("m2", calculadora.getResultado\_area().unidaddemedida);

}

}

}

## Utilizado desde un formulario Windows Forms

using System;

using System.Windows.Forms;

namespace Pruebas\_FORMS\_Core

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

CalculoSimple.CalculoSimple calculadora = new CalculoSimple.CalculoSimple();

try

{

calculadora.setDato\_unlado(new Magnitudes.Magnitud(Double.Parse(txtUnLado\_valor.Text),

txtUnLado\_unidaddemedida.Text));

calculadora.setDato\_otrolado(new Magnitudes.Magnitud(Double.Parse(txtOtroLado\_valor.Text),

txtOtroLado\_unidaddemedida.Text));

}

catch (ArgumentOutOfRangeException ex)

{

MessageBox.Show("Alguna de las unidades de medida no está contemplada en las conversiones: " + ex.Message);

}

catch (System.IO.FileNotFoundException ex)

{

MessageBox.Show("No se ha encontrado el archivo con la lista de conversiones entre unidades de medida: " + ex.Message);

}

catch (ArgumentException ex)

{

MessageBox.Show("Problemas leyendo el archivo con la lista de conversiones entre unidades de medida: " + ex.Message);

}

catch (Exception ex)

{

MessageBox.Show("Problemas en alguna conversión entre unidades de medida: " + ex.Message);

}

txtPerimetro.Text = calculadora.getResultado\_perimetro().ToString();

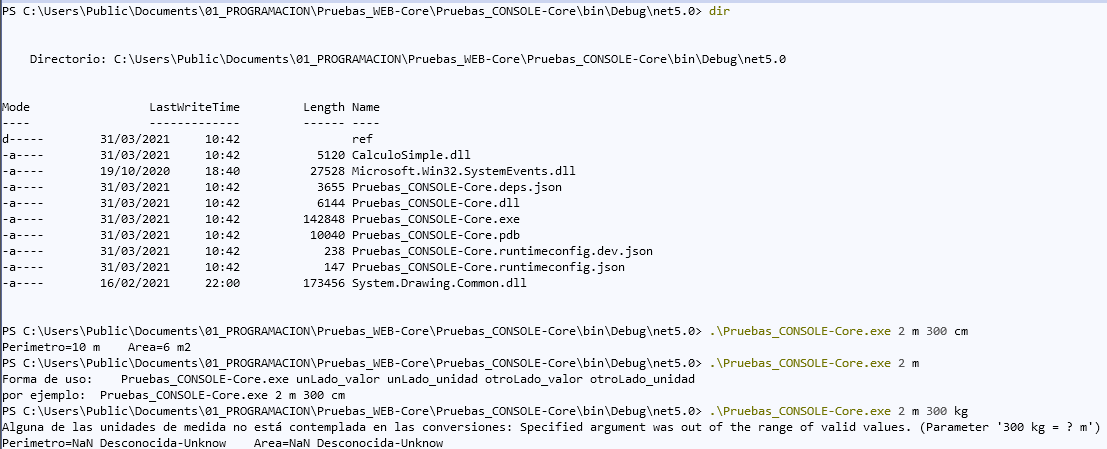
txtArea.Text = calculadora.getResultado\_area().ToString();

}

}

}

## Utilizado desde linea de comandos

using System;

namespace Pruebas\_CONSOLE\_Core

{

class Program

{

static void Main(string[] args)

{

if (args.Length != 4)

{

Console.WriteLine("Forma de uso: Pruebas\_CONSOLE-Core.exe unLado\_valor unLado\_unidad otroLado\_valor otroLado\_unidad");

Console.WriteLine("por ejemplo: Pruebas\_CONSOLE-Core.exe 2 m 300 cm");

}

else

{

CalculoSimple.CalculoSimple calculadora = new CalculoSimple.CalculoSimple();

try

{

calculadora.setDato\_unlado(new Magnitudes.Magnitud(Double.Parse(args[0]), args[1]));

calculadora.setDato\_otrolado(new Magnitudes.Magnitud(Double.Parse(args[2]), args[3]));

}

catch (ArgumentOutOfRangeException ex)

{

Console.WriteLine("Alguna de las unidades de medida no está contemplada en las conversiones: " + ex.Message);

}

catch (System.IO.FileNotFoundException ex)

{

Console.WriteLine("No se ha encontrado el archivo con la lista de conversiones entre unidades de medida: " + ex.Message);

}

catch (ArgumentException ex)

{

Console.WriteLine("Problemas leyendo el archivo con la lista de conversiones entre unidades de medida: " + ex.Message);

}

catch (Exception ex)

{

Console.WriteLine("Problemas en alguna conversión entre unidades de medida: " + ex.Message);

}

Console.WriteLine("Perimetro=" + calculadora.getResultado\_perimetro().ToString()

+ " Area=" + calculadora.getResultado\_area().ToString());

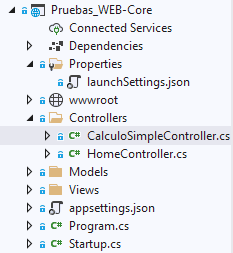
}

}

}

}

## Utilizado para dar unos servicios web RESTfull



using Microsoft.AspNetCore.Mvc;

namespace Pruebas\_WEB\_Core.Controllers

{

public class ParametrosDelCalculoSimple

{

public Magnitudes.Magnitud unLado { get; set; }

public Magnitudes.Magnitud otroLado { get; set; }

}

public class ResultadosDelCalculoSimple

{

public Magnitudes.Magnitud perimetro { get; set; }

public Magnitudes.Magnitud area { get; set; }

}

[ApiController]

[Route("api/[controller]")]

public class CalculoSimpleController : ControllerBase

{

// nota: por URL, los parametros se pasan algo así como:

// https://localhost:44310/api/CalculoSimple?unLado\_valor=2&unLado\_unidaddemedida=m&otroLado\_valor=300&otroLado\_unidaddemedida=cm

[HttpGet]

public ResultadosDelCalculoSimple Get(double unLado\_valor, string unLado\_unidaddemedida,

double otroLado\_valor, string otroLado\_unidaddemedida)

{

CalculoSimple.CalculoSimple calculadora = new CalculoSimple.CalculoSimple();

calculadora.setDato\_unlado(new Magnitudes.Magnitud(unLado\_valor, unLado\_unidaddemedida));

calculadora.setDato\_otrolado(new Magnitudes.Magnitud(otroLado\_valor, otroLado\_unidaddemedida));

ResultadosDelCalculoSimple resultados = new ResultadosDelCalculoSimple();

resultados.perimetro = calculadora.getResultado\_perimetro();

resultados.area = calculadora.getResultado\_area();

return resultados;

}

// nota: en JSON el resultado se devuelve algo así como:

// {

// "perimetro": {

// "valor": xxxxx

// , "unidaddemedida": "xxxxxx"

// }

// , "area": {

// "valor": xxxxx

// , "unidaddemedida": "xxxxxxx"

// }

// }

//Escribiendolo en forma compacta: { "perimetro":{ "valor":10,"unidaddemedida":"m"},"area":{ "valor":6,"unidaddemedida":"m2"} }

// nota: en JSON hay que pasar los parametros algo asi como:

// {

// "unLado": {

// "valor": xxxxx

// , "unidaddemedida":"xxxxxxx"

// }

// , "otroLado": {

// "valor": xxxxx

// , "unidaddemedida":"xxxxxxx"

// }

// }

//Escribiendolo en forma compacta: {"unLado":{"valor":2,"unidaddemedida":"m"},"otroLado":{ "valor":300,"unidaddemedida":"cm"}}

[HttpPost]

public ResultadosDelCalculoSimple Post([FromBody] ParametrosDelCalculoSimple parametros)

{

CalculoSimple.CalculoSimple calculadora = new CalculoSimple.CalculoSimple();

calculadora.setDato\_unlado(parametros.unLado);

calculadora.setDato\_otrolado(parametros.otroLado);

ResultadosDelCalculoSimple resultados = new ResultadosDelCalculoSimple();

resultados.perimetro = calculadora.getResultado\_perimetro();

resultados.area = calculadora.getResultado\_area();

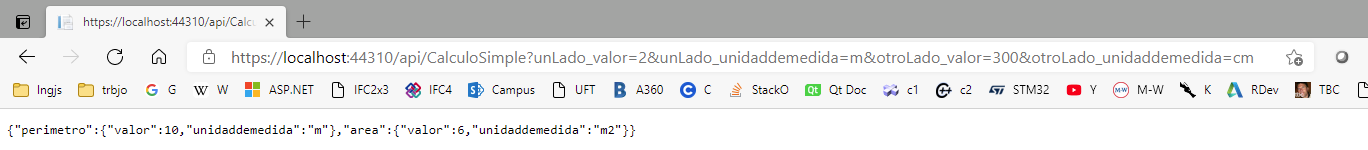
return resultados;

}

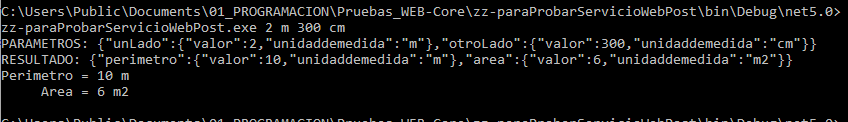
}

}

### Utilizando el servicio HttpGet a través de una url



### Utilizando el servicio HttpPut a través de un programilla de linea de comando



using System;

namespace zz\_paraProbarServicioWebPost

{

public class Magnitud

{

public double valor { get; set; }

public string unidaddemedida { get; set; }

}

public class ParametrosDelCalculoSimple

{

public Magnitud unLado { get; set; }

public Magnitud otroLado { get; set; }

}

public class ResultadosDelCalculoSimple

{

public Magnitud perimetro { get; set; }

public Magnitud area { get; set; }

}

class Program

{

static async System.Threading.Tasks.Task Main(string[] args)

{

System.Net.Http.HttpClient clienteWeb = new System.Net.Http.HttpClient();

if (args.Length != 4)

{

Console.WriteLine("Forma de uso: zz-paraProbarServicioWebPost.exe unLado\_valor unLado\_unidad otroLado\_valor otroLado\_unidad");

Console.WriteLine("por ejemplo: paraProbarServicioWebPost.exe 2 m 300 cm");

}

else

{

Magnitud unLado = new Magnitud();

unLado.valor = double.Parse(args[0]);

unLado.unidaddemedida = args[1];

Magnitud otroLado = new Magnitud();

otroLado.valor = double.Parse(args[2]);

otroLado.unidaddemedida = args[3];

ParametrosDelCalculoSimple parametros = new ParametrosDelCalculoSimple();

parametros.unLado = unLado;

parametros.otroLado = otroLado;

string parametrosJson = System.Text.Json.JsonSerializer.Serialize(parametros);

Console.WriteLine("PARAMETROS: " + parametrosJson);

try

{

System.Net.Http.HttpContent parametrosCodificados =

new System.Net.Http.StringContent(content: parametrosJson,

encoding: System.Text.Encoding.UTF8,

mediaType: "application/json");

System.Net.Http.HttpResponseMessage respuesta = await clienteWeb.PostAsync("https://localhost:44310/api/CalculoSimple",

parametrosCodificados);

respuesta.EnsureSuccessStatusCode();

string resultadoEnBruto = await respuesta.Content.ReadAsStringAsync();

Console.WriteLine("RESULTADO: " + resultadoEnBruto);

ResultadosDelCalculoSimple resultados =

(ResultadosDelCalculoSimple)System.Text.Json.JsonSerializer.Deserialize(resultadoEnBruto,

typeof(ResultadosDelCalculoSimple));

Console.WriteLine("Perimetro = " + resultados.perimetro.valor.ToString() + " " + resultados.perimetro.unidaddemedida);

Console.WriteLine(" Area = " + resultados.area.valor.ToString() + " " + resultados.area.unidaddemedida);

}

catch (Exception ex)

{

Console.WriteLine("ERROR: " + ex.Message);

}

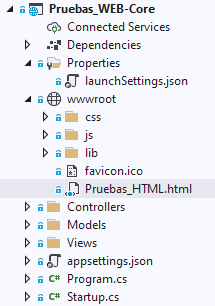
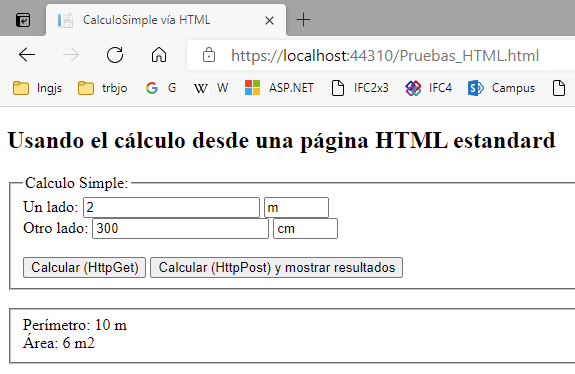
}

}

}

}

### Utilizando el servicio HttpPut desde una página web HTML/JavaScript

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title>CalculoSimple vía HTML</title>

</head>

<body>

<h2>Usando el cálculo desde una página HTML estandard</h2>

<div>

<form name="frmCalculoSimple" action="/api/CalculoSimple" target="\_self">

<fieldset>

<legend>Calculo Simple:</legend>

<label for="unLado\_valor unLado\_unidaddemedida">Un lado: </label>

<input type="number" id="unLado\_valor" name="unLado\_valor" required value="2" />

<input type="text" id="unLado\_unidaddemedida" name="unLado\_unidaddemedida" size="4" placeholder="cm" required value="m" />

<br />

<label for="otroLado\_valor otroLado\_unidaddemedida">Otro lado: </label>

<input type="number" id="otroLado\_valor" name="otroLado\_valor" required value="300" />

<input type="text" id="otroLado\_unidaddemedida" name="otroLado\_unidaddemedida" size="4" placeholder="cm" required value="cm"/>

<br />

<br />

<input type="submit" value="Calcular (HttpGet)" formmethod="get" />

<button type="button" onclick="CalcularYMostrarResultados()">Calcular (HttpPost) y mostrar resultados</button>

</fieldset>

<br />

<fieldset>

<label for="perimetro">Perímetro: </label>

<output id="perimetro"></output>

<br />

<label for="area">Área: </label>

<output id="area"></output>

</fieldset>

</form>

</div>

<script>

function CalcularYMostrarResultados() {

var solicitud = new XMLHttpRequest();

solicitud.onload = function () {

var respuesta = solicitud.responseText;

if (solicitud.statusText = "OK") {

var resultados = JSON.parse(respuesta)

document.getElementById("perimetro").value = resultados.perimetro.valor + " " + resultados.perimetro.unidaddemedida;

document.getElementById("area").value = resultados.area.valor + " " + resultados.area.unidaddemedida;

}

}

var esAsincrono = true;

solicitud.open("POST", "/api/CalculoSimple", esAsincrono);

solicitud.setRequestHeader("Content-Type", "application/json;charset=UTF-8");

var parametros = JSON.stringify(

{

"unLado": {

"valor": document.getElementById("unLado\_valor").value

, "unidaddemedida": document.getElementById("unLado\_unidaddemedida").value

}

, "otroLado": {

"valor": document.getElementById("otroLado\_valor").value

, "unidaddemedida": document.getElementById("otroLado\_unidaddemedida").value

}

}

)

solicitud.send(parametros);

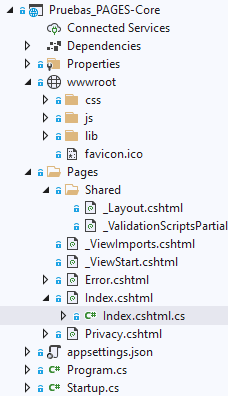
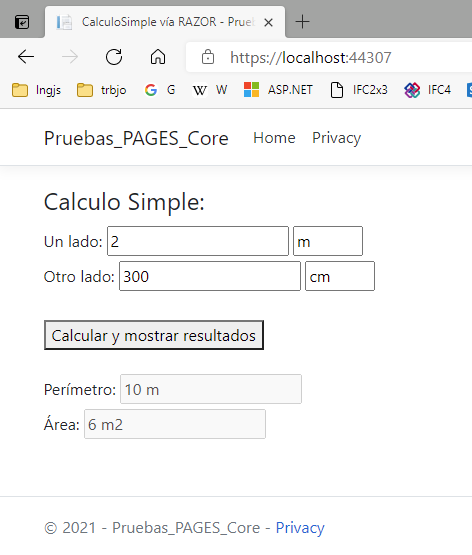
}

</script>

</body>

</html>

## Utilizado desde una página web RAZOR/C#

### index.cshtml

@page

@model IndexModel

@{

ViewData["Title"] = "CalculoSimple vía RAZOR";

}

<div>

<form method="post">

<fieldset>

<legend>Calculo Simple:</legend>

<label for="unLado\_valor unLado\_unidaddemedida">Un lado: </label>

<input type="number" asp-for="unLado\_valor" required />

<input type="text" asp-for="unLado\_unidaddemedida" size="4" placeholder="cm" required />

<br />

<label for="otroLado\_valor otroLado\_unidaddemedida">Otro lado: </label>

<input type="number" asp-for="otroLado\_valor" required />

<input type="text" asp-for="otroLado\_unidaddemedida" size="4" placeholder="cm" required />

<br />

<br />

<button type="submit">Calcular y mostrar resultados</button>

</fieldset>

<br />

<fieldset>

<label for="perimetro">Perímetro: </label>

<input asp-for="perimetro" disabled />

<br />

<label for="area">Área: </label>

<input asp-for="area" disabled />

</fieldset>

</form>

</div>

### Index.cshtml.cs

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.RazorPages;

using Microsoft.Extensions.Logging;

using System;

namespace Pruebas\_PAGES\_Core.Pages

{

public class IndexModel : PageModel

{

[BindProperty]

public double unLado\_valor { get; set; }

[BindProperty]

public string unLado\_unidaddemedida { get; set; }

[BindProperty]

public double otroLado\_valor { get; set; }

[BindProperty]

public string otroLado\_unidaddemedida { get; set; }

[BindProperty]

public string perimetro { get; private set; }

[BindProperty]

public string area { get; private set; }

public IActionResult OnGet()

{

unLado\_valor = 2;

unLado\_unidaddemedida = "m";

otroLado\_valor = 300;

otroLado\_unidaddemedida = "cm";

return Page();

}

public IActionResult OnPost()

{

CalculoSimple.CalculoSimple calculadora = new CalculoSimple.CalculoSimple();

try

{

calculadora.setDato\_unlado(new Magnitudes.Magnitud(unLado\_valor, unLado\_unidaddemedida));

calculadora.setDato\_otrolado(new Magnitudes.Magnitud(otroLado\_valor, otroLado\_unidaddemedida));

}

catch (ArgumentOutOfRangeException ex)

{

//MessageBox.Show("Alguna de las unidades de medida no está contemplada en las conversiones: " + ex.Message);

}

catch (System.IO.FileNotFoundException ex)

{

//MessageBox.Show("No se ha encontrado el archivo con la lista de conversiones entre unidades de medida: " + ex.Message);

}

catch (ArgumentException ex)

{

//MessageBox.Show("Problemas leyendo el archivo con la lista de conversiones entre unidades de medida: " + ex.Message);

}

catch (Exception ex)

{

//MessageBox.Show("Problemas en alguna conversión entre unidades de medida: " + ex.Message);

}

perimetro = calculadora.getResultado\_perimetro().ToString();

area = calculadora.getResultado\_area().ToString();

return Page();

}

}

}