

**Universidad Abierta y a Distancia**

**de México**

**División de Ciencias Exactas, Ingeniería y Tecnología**

**Desarrollo de software**

**Semestre:** 5to

**Asignatura:** Programación net I

**Unidad de aprendizaje:** 2

**Actividad: Evidencia de aprendizaje**

**Nombre del estudiante:** Francisco Manuel López García

**Matrícula:** ES1921010154

**Grupo:** DS-DPRN1-2202-B1-001

**Docente:** Jacobo Díaz Martínez

**Fecha de entrega:** 29 de Julio 2022

**Ciudad de México, julio del 2022**

Diseño: DL-CPL

**ÍNDICE**

[**INTRODUCCIÓN** 3](#_Toc80333573)

[**DESARROLLO DE LA ACTIVIDAD** 4](#_Toc80333574)

[**CONCLUSIONES** 12](#_Toc80333575)

[**REFERENCIAS BIBLIOGRÁFICAS** 12](#_Toc80333576)

# **INTRODUCCIÓN**

En esta actividad desarrollaremos un programa en el lenguaje C# el cual se trata de una casa de cambio “El Conversor” te ha contratado pues su antiguo programador renunció, necesita que des seguimiento a un programa que administre las transacciones de cambio de divisa. Para su desarrollo se hace uso de una API que nos muestra los montos de cambio de divisas, al desarrollar nos otorgan un código incompleto, el cual mejoramos para que e programa sea eficiente. Para ello leemos la documentación de la API y hacemos uso de un EndPoint que nos retorna el cambio de divisa, se le tiene que mandar como parámetro el tipo de divisa base, la divisa de cambio y el monto a cambiar.  
Como mencioné en la actividad anterior convertí mi menú interactivo en una clase en un archivo separado el cual contiene un menú que nos retorna posición y un menú que nos retorna un string.

# **DESARROLLO DE LA ACTIVIDAD**

Código fuente:  
*Modelo.cs (Clase Modelo que es el modelo de la respuesta de la API)*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

//Modelo de la response de nuestra API

namespace modelo

{

public class Query

{

public string from { get; set; }

public string to { get; set; }

public double amount { get; set; }

}

public class Info

{

public int timestamp { get; set; }

public double rate { get; set; }

}

public class Root

{

public string @base { get; set; }

public bool success { get; set; }

public Query query { get; set; }

public Info info { get; set; }

public string date { get; set; }

public double result { get; set; }

}

}

*menuInteractivo.cs (Clase Menu que es el menú interactivo)*

/\* ---------- Ayuda!! ----------

\* Se muestra un menú interactivo que utiliza las teclas arriba y abajo para moverte en el menú,

\* la tecla Enter para seleccionar.

\* Para ello se utiliza un ciclo doWhile donde con un for y un variable opcionSeleccionada que se aumenta

\* o disminuye según se pulsa abajo o arriba, se muestra el menú, cambiando de color la opción que

\* se está seleccionando. Este doWhile se repite hasta que se presione la tecla Enter

\*/

//-Importaciones:

using System;

//-Contenido:

namespace menus

{

class Menu{

public Menu()

{

//constructor vació

}

// Método de menú con Titulo, indicación y arreglo de opciones, que devuelve el entero representando la posición

public static int menuToPosition(string titulo, string indicacion, string[] opciones){

int opcionSeleccionada = 0;// Variable para guardar la opción del menú en que nos encontramos

ConsoleKey key = new ConsoleKey(); // Variable para almacenar la tecla pulsada

do{

Console.Clear(); // Limpia la consola

Console.ForegroundColor = ConsoleColor.DarkCyan; // Cambia el color del texto

Console.WriteLine(titulo);

Console.ForegroundColor = ConsoleColor.DarkBlue; // Cambia el color del texto

Console.WriteLine(indicacion);

Console.ForegroundColor = ConsoleColor.DarkGray; // Cambia el color del texto

Console.WriteLine("Utilize las flechas para moverse y pulse Enter para seleccionar... \n");

// for que muestra las opciones poniendo de color blanco en donde se está posicionado

for(int x = 0 ; x < opciones.Length ; x++){

if(opcionSeleccionada == x){

Console.ForegroundColor = ConsoleColor.White; // Cambia el color del texto

Console.WriteLine(opciones[x]);

}else {

Console.ForegroundColor = ConsoleColor.DarkYellow; // Cambia el color del texto

Console.WriteLine(opciones[x]);

}

}

// Se obtiene la tecla pulsada

key = Console.ReadKey().Key;

// Se evalúa si se pulso arriba o abajo

if(key == ConsoleKey.UpArrow){

// Se asigna a la variable opcionSeleccionada un decremento siempre y cuando el valor no sea 0

// Si es 0 se asigna un la última opcionSeleccionada que sería la última opcionSeleccionada del menú

opcionSeleccionada = opcionSeleccionada == 0 ? opciones.Length-1 : opcionSeleccionada - 1 ;

} else if(key == ConsoleKey.DownArrow){

// Se asigna a la variable opcionSeleccionada un incremento siempre y cuando el valor no sea 2

// Si es la última opcionSeleccionada se asigna un 0 que sería la primera opcionSeleccionada del menú

opcionSeleccionada = opcionSeleccionada == opciones.Length-1 ? 0 : opcionSeleccionada + 1 ;

}

}while(key != ConsoleKey.Enter);

return opcionSeleccionada;

}

// Método de menú con Titulo, indicación y arreglo de opciones, que devuelve el string seleccionado

public static string menuToString(string titulo, string indicacion, string[] opciones){

int opcionSeleccionada = 0;// Variable para guardar la opción del menú en que nos encontramos

ConsoleKey key = new ConsoleKey(); // Variable para almacenar la tecla pulsada

do{

Console.Clear(); // Limpia la consola

Console.ForegroundColor = ConsoleColor.DarkCyan; // Cambia el color del texto

Console.WriteLine(titulo);

Console.ForegroundColor = ConsoleColor.DarkBlue; // Cambia el color del texto

Console.WriteLine(indicacion);

Console.ForegroundColor = ConsoleColor.DarkGray; // Cambia el color del texto

Console.WriteLine("Utilize las flechas para moverse y pulse Enter para seleccionar... \n");

// for que muestra las opciones poniendo de color blanco en donde se está posicionado

for(int x = 0 ; x < opciones.Length ; x++){

if(opcionSeleccionada == x){

Console.ForegroundColor = ConsoleColor.White; // Cambia el color del texto

Console.WriteLine(opciones[x]);

}else {

Console.ForegroundColor = ConsoleColor.DarkYellow; // Cambia el color del texto

Console.WriteLine(opciones[x]);

}

}

// Se obtiene la tecla pulsada

key = Console.ReadKey().Key;

// Se evalúa si se pulso arriba o abajo

if(key == ConsoleKey.UpArrow){

// Se asigna a la variable opcionSeleccionada un decremento siempre y cuando el valor no sea 0

// Si es 0 se asigna un la última opcionSeleccionada que sería la última opcionSeleccionada del menú

opcionSeleccionada = opcionSeleccionada == 0 ? opciones.Length-1 : opcionSeleccionada - 1 ;

} else if(key == ConsoleKey.DownArrow){

// Se asigna a la variable opcionSeleccionada un incremento siempre y cuando el valor no sea 2

// Si es la última opcionSeleccionada se asigna un 0 que sería la primera opcionSeleccionada del menú

opcionSeleccionada = opcionSeleccionada == opciones.Length-1 ? 0 : opcionSeleccionada + 1 ;

}

}while(key != ConsoleKey.Enter);

return opciones[opcionSeleccionada];

}

}

}

*Program.cs (Clase Program que es la clase principal)*

/\* ---------- Ayuda!! ----------

\* Caso de estudio. Evidencia de aprendizaje

\*/

//-Importaciones:

using RestSharp;

using Newtonsoft.Json;

using System;

using modelo; // Modelo de la response de la API

using menus; // importación para menús interactivos

//-Contenido:

class Program

{

// Método para consultar API y realizar el cambio de divisa

static Root cambioDivisaAPI(string divisaBase, string divisaCambio, double montoACambiar){

var client = new RestClient("https://api.apilayer.com/exchangerates\_data/convert"); //URL de la API

RestRequest request = new RestRequest();

request.AddParameter("from",divisaBase); // Se agrega el parámetro from con la divisa base

request.AddParameter("to",divisaCambio); // Se agrega el parámetro to con la divisa de cambio

request.AddParameter("amount",montoACambiar); // Se agrega el parámetro amount con el monto a cambiar

request.AddHeader("apikey", "3pXSyZ8O0C07liPcjUM57O0KDy99W8As");//Se añade en el header de la consulta la APIKey

var response = client.Execute(request);

Root respuesta = JsonConvert.DeserializeObject<Root>(response.Content); // Se deserializa el JSON de la response y se asigna al objeto Root

return respuesta; // Se retorna el objeto de la clase Root

}

// Método para calcular comisión

static double calcularComision(double montoFinal){

return (montoFinal \* .05); // se retorna el 5% del monto final

}

// Método para mostrar resultado

static void infoTransaccion(string divisaBase, string divisaCambio, double montoACambiar, double montoFinal, double comision, int timestamp){

// Se crea una variable de tipo DateTime para la hora de la transacción

DateTime horaTransaccion = new DateTime();

// Se le asigna la hora a partir del timestamp de la API y se llama al método ToLocalTime para convertirlo a la zona horaria actual

horaTransaccion = horaTransaccion.AddSeconds(timestamp).ToLocalTime();

Console.ForegroundColor = ConsoleColor.DarkBlue; // Cambia el color del texto

Console.WriteLine("- Información de la transacción: \n");

Console.ForegroundColor = ConsoleColor.DarkGreen; // Cambia el color del texto

Console.WriteLine("Divisa base: $" + montoACambiar.ToString("N2") + " " + divisaBase);

Console.WriteLine("Divisa de cambio: $" + montoFinal.ToString("N2") + " " + divisaCambio);

Console.WriteLine("Comisión: $" + comision.ToString("N2") + " " + divisaCambio);

Console.WriteLine("Hora de transacción: " + horaTransaccion.ToString("hh:mm:ss"));

}

// Método Main principal

static void Main(string[] args)

{

// Creación de variables

Root respuesta = new Root();

string titulo = "----- E L C O N V E R S O R -----";

string divisaBase, divisaCambio;

double comision;

double montoACambiar, montoFinal;

// Se crea un arreglo con las opciones de nuestro menú interactivo

string[] divisas = {

"USD",

"MXN",

"EUR",

"BTC"

};

// Se manda llamar nuestro menú, con titulo, indicación y opciones. El valor de retorno se asigna a la divisaBase

divisaBase = Menu.menuToString(titulo,"? Divisa base:", divisas);

// Se manda llamar nuestro menú, con titulo, indicación y opciones. El valor de retorno se asigna a la divisaCambio

divisaCambio = Menu.menuToString(titulo,"? Divisa cambio:", divisas);

//Se pide el monto a cambiar

Console.Clear(); // Limpia la consola

Console.ForegroundColor = ConsoleColor.DarkCyan; // Cambia el color del texto

Console.WriteLine(titulo);

Console.ForegroundColor = ConsoleColor.DarkBlue; // Cambia el color del texto

Console.WriteLine("? Monto a cambiar:");

Console.ForegroundColor = ConsoleColor.Gray; // Cambia el color del texto

montoACambiar = double.Parse(Console.ReadLine());// Se parsea a double lo que escribe el usuario para guardarla en la variable

// Se realiza el cambio de divisa con una consulta a la API

respuesta = cambioDivisaAPI(divisaBase, divisaCambio, montoACambiar);

montoFinal = respuesta.result; // se asigna el resultado al montoFinal

comision = calcularComision(montoFinal); // Se calcula la comisión a partir del montoFinal

//Se muestran los resultados

Console.Clear(); // Limpia la consola

Console.ForegroundColor = ConsoleColor.DarkCyan; // Cambia el color del texto

Console.WriteLine(titulo);

// Se llama al método que se encarga de mostrar todos los resultados

infoTransaccion(divisaBase, divisaCambio, montoACambiar, montoFinal, comision, respuesta.info.timestamp);

Console.ForegroundColor = ConsoleColor.Gray; // Cambia el color del texto

// Se pausa el programa en espera de que se pulse cualquier tecla para finalizarlo

Console.ReadKey();

}

}

Capturas de pantalla:

Para la ejecución del programa se utilizaron dos ejemplos. El primero de ellos:

Divisa base: USD, Divisa cambio: MXN, Monto: $1478

Ejecución del programa:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente con confianza media

El segundo de ellos:

Divisa base: MXN, Divisa cambio: EUR, Monto: $52450

Ejecución del programa:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente con confianza media

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente con confianza baja

Texto

Descripción generada automáticamente

# **CONCLUSIONES**

En esta actividad pude entender más el uso de API’s que tenemos a nuestro alcance en la web, utilizando la documentación pude comprender los EndPoints que nos otorga esta API y así utilizar el que mas se adecue al programa generado. Saber utilizar la API mediante su documentación nos aporta un conocimiento amplio de las opciones que nos ofrece, es d es mucha utilidad y yo diría necesario conocer las documentaciones.  
A su vez, utilice el objeto que nos retorna para poder mostrar los resultados requeridos, se hizo uso de un timestamp que nos regresaba la API para obtener la hora de la transacción.  
Implemente el menú interactivo como una clase separada, que me servirá para reutilizarlo en cualquier proyecto de consola que requiera de un menú, sin importar la cantidad de opciones que tenga dicho menú, así como poder optar entre un menú que nos retorne la posición o el string seleccionado, con lo cual se vuelve un menú robusto y escalable.

# **REFERENCIAS BIBLIOGRÁFICAS**

Ceballos, F (2007). Enciclopedia de Microsoft Visual C#, (2da. Ed.). España: Ed. Alfaomega Ra-MA.

MSDN Microsoft (2012), Guía de programación en C#, recuperado de la siguiente dirección: <https://www.lawebdelprogramador.com/pdf/8631-C-Guia-Total-delProgramador.html>

Shart, J & Jagger, J (2002). Microsoft VISUAL C#. NET, APRENDA YA. España: Mc Graw Hill.