Herramientas de Programación 1

Tema Nº13:

Consumir servicios web REST desde aplicaciones de escritorio.

Indicador de logro Nº13:Desarrolla aplicaciones de escritorio con consumo de servicios web REST haciendo uso de librerías .NET.

**TEMA 01 Teoría de los**

Imagen que contiene Icono

Descripción generada automáticamente

**TEMA Nº13:**

CONSUMIR SERVICIOS WEB REST DESDE APLICACIONES DE ESCRITORIO. PARTE 1.

Realizar consultas, mediante una aplicación de escritorio Windows, a un Servicio Web Rest de Sunat.

Este servicio web no permitirá obtener un Token, un Token es algo funcionalmente parecido a una credencial que nos permitirá realizar otras consultas, en nuestro caso, a consultar la validez de un comprobante de pago.

Para generar un Token es necesario:

* Estar registrado en Sunat.
* Ingresar a Sunat con la Clave Sol y generar un Usuario y Contraseña (client\_id, client\_secret)

El procedimiento para crear ambos datos y otros detalles necesarios lo podemos ver en el manual cuyo link mostramos en la sección Referencias de esta guía.

Restsharp

En resumen, RestSharp es una librería de .NET que nos permite consumir Servicios Web Rest, e inclusive crearlos. En nuestros ejemplos veremos solo algunas opciones.

RestSharp es una biblioteca de cliente HTTP integral de código abierto que funciona con todo tipo de tecnologías .Net. Se puede usar para crear aplicaciones robustas al facilitar la interfaz con las API públicas y acceder rápidamente a los datos sin la complejidad de manejar solicitudes HTTP sin procesar. RestSharp combina innumerables ventajas y características que ahorran tiempo con una interfaz simple y limpia, lo que la convierte en una de las herramientas REST más populares que se utilizan en la actualidad.

Con su API simple y su potente biblioteca, la arquitectura REST es la herramienta elegida por los programadores que buscan crear programas y aplicaciones detallados. La arquitectura RESTful proporciona un enfoque orientado a los recursos y orientado a la información para crear aplicaciones web. También ofrece tareas comunes, como la generación de URI, el análisis de la carga útil y la autenticación, como opciones configurables, lo que garantiza que los desarrolladores de aplicaciones ya no tengan que preocuparse por tareas de bajo nivel, como las redes.

RestSharp es una de las mejores bibliotecas para usar si usa REST con frecuencia para consumir API HTTP en .Net. Resulta especialmente útil para las aplicaciones de Windows, donde REST o SOAP a menudo se utilizan para comunicarse con datos externos.

El manejo de solicitudes asíncronas es uno de los requisitos más importantes para programar en plataformas Windows. RestSharp admite solicitudes síncronas y asíncronas, lo que lo hace perfecto para aplicaciones de Windows. Esta potente biblioteca ahorra tiempo de programación y equipa a los desarrolladores con herramientas útiles que ayudan a crear aplicaciones elegantes que son fáciles de depurar.

**Ejemplos:**

**RECURSOS**

1. Hardware

* Una computadora con 4 GB e RAM y procesador de 2.0 GHz.

1. Software

* Visual Studio Professional 2017 / 2019
* RestSharp

**PROCEDIMIENTO**

**Laboratorio 13:**

13.1 Consultar el servicio web Rest de Sunat que nos devuelve el Token.

**Objetivos**

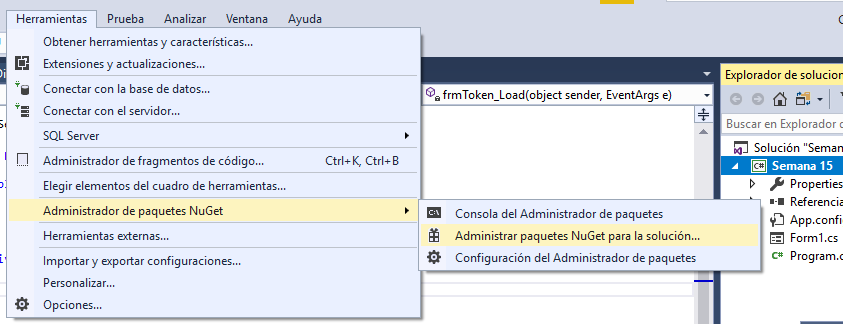
* Agregar la librería RestSharp al proyecto.
* Utilizar RestSharp para consumir el servicio web Rest de Sunat que devuelve el Token.

**Introducción:**

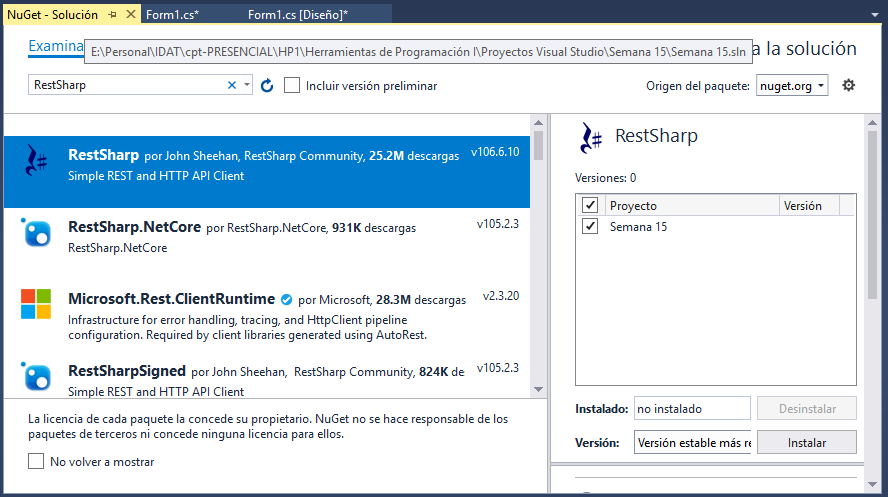
* Para consultar los servicios web Rest de Sunat vamos a utilizar la librería RestSharp, por lo que antes de consultar los servicios web debemos agregar RestSharp a nuestro proyecto.

Paso 1. Preparar el entorno.

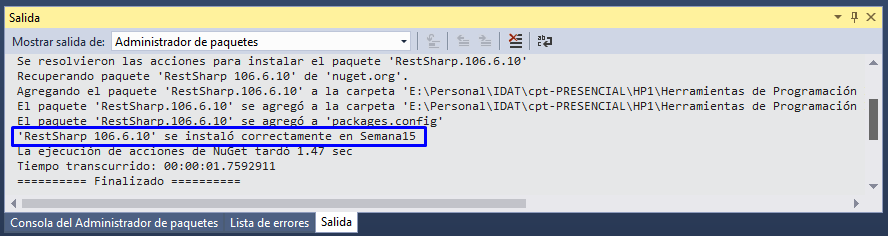
1. Creamos el proyecto Semana13.
2. Para agregar RestSharp al proyecto utilizamos el Administrador de paquetes NuGet



1. Buscamos RestSharp y lo instalamos.

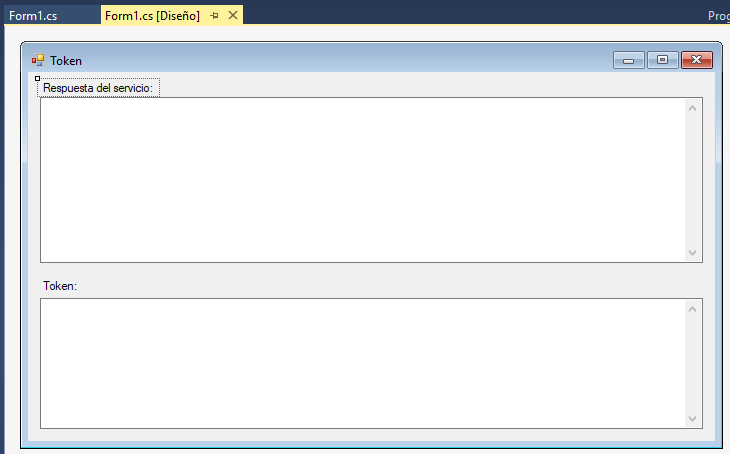


1. Validamos que la librería se instaló correctamente.



Paso 2. Consumir el servicio web Rest que devuelve el Token.

1. Diseñamos el formulario de acuerdo a:



|  |  |  |
| --- | --- | --- |
| Control | Propiedad | Valor |
| Form1 | (Name) | frmToken |
|  | Text | Token |
| TextBox1 | (Name) | txtResponse |
|  | Multiline | True |
|  | ScrollBars | Vertical |
| TextBox2 | (Name) | txtToken |
|  | Multiline | True |
|  | ScrollBars | Vertical |

1. Creamos la clase “Constante” en el archivo “Constante.cs”.

Esta clase almacenara valores constantes que usaremos en la consulta del servicio web.

Definición de la clase:

public static class Constantes

{

public const string URI\_TOKEN = "https://api-seguridad.sunat.gob.pe/v1/clientesextranet/ce58910f-7676-4431-8019-63b3a07f640d/oauth2/token/";

public const string URI\_CONSULTA = "https://api.sunat.gob.pe/v1/contribuyente/contribuyentes/10161684916/validarcomprobante";

public const string CONTENT\_TYPE = "application/x-www-form-urlencoded";

public const string GRANT\_TYPE = "client\_credentials";

public const string SCOPE = "https://api.sunat.gob.pe/v1/contribuyente/contribuyentes";

public const string CLIENT\_ID = "ce58910f-7676-4431-8019-63b3a07f640d";

public const string CLIENT\_SECRET = "w1ZtI/KoQEc7yIgkEW2zSA==";

}

1. Creamos la clase “JsonResponseToken” en el archivo “JsonResponseToken.cs”.

Esta clase se asociara al JSon de la respuesta de la consulta (response).

Definición de la clase:

public class JsonResponseToken

{

public string Access\_token { get; set; }

public string Token\_type { get; set; }

public int Expires\_in { get; set; }

}

1. Agregamos código al evento Load del formulario de acuerdo a:

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

{

var clienteRest = new RestClient(Constantes.URI\_TOKEN);

var request = new RestRequest(Method.POST);

request.AddHeader("Content-Type", Constantes.CONTENT\_TYPE);

request.AddParameter("grant\_type", Constantes.GRANT\_TYPE);

request.AddParameter("scope", Constantes.SCOPE);

request.AddParameter("client\_id", Constantes.CLIENT\_ID);

request.AddParameter("client\_secret", Constantes.CLIENT\_SECRET);

IRestResponse restResponseToken = clienteRest.Execute(request);

txtResponse.Text = restResponseToken.Content;

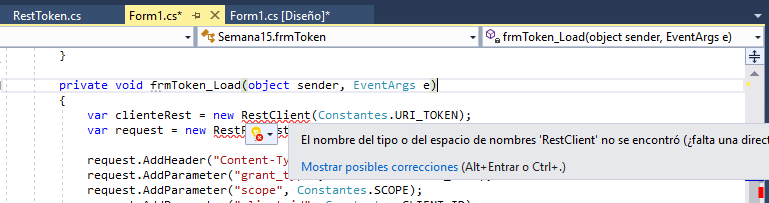
JsonDeserializer deserial = new JsonDeserializer();

JsonResponseToken jsonResponseToken = deserial.Deserialize<JsonResponseToken>(restResponseToken);

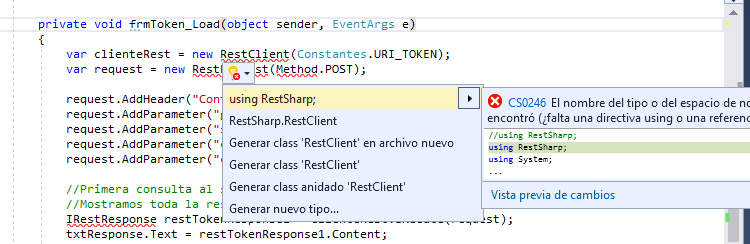
txtToken.Text = jsonResponseToken.Access\_token;

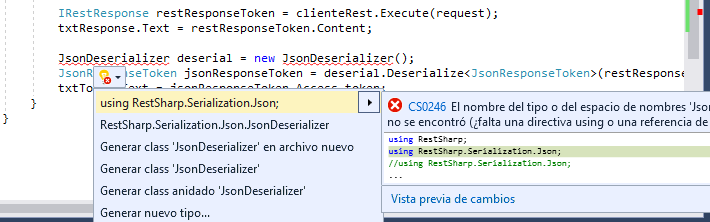
}

1. Se van a mostrar errores, esto es debido a que no hemos hecho referencia a las librerías de RestSharp.

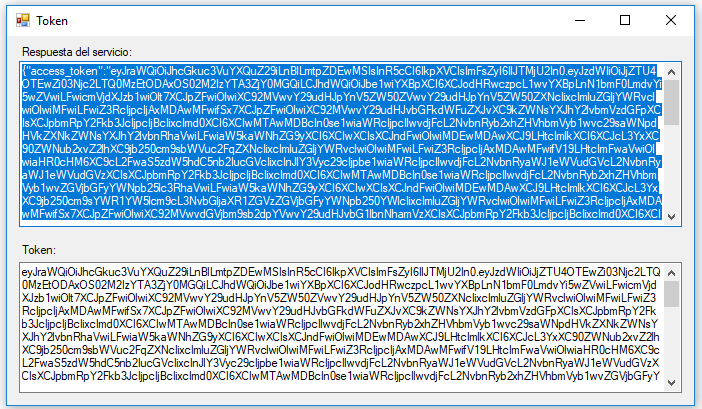


1. Hacemos click en el enlace “Mostrar posibles correcciones” y seleccionamos “using RestSharp”





1. Ejecutamos la aplicación y validamos que se obtiene la respuesta de la consulta y el valor del Token.



**Actividad:**

Ingresa a la plataforma virtual, revisa y analiza los siguientes enlaces, luego desarrolla las actividades propuestas:

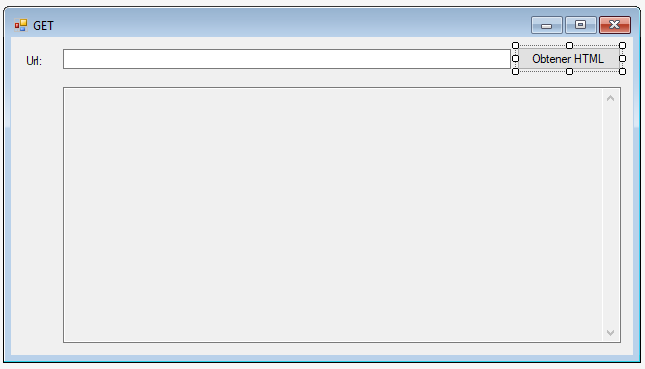
* RestSharp

<https://stackify.com/restsharp/>

* Sunat: Manual de Consulta de Comprobantes de Pago por Servicio Web

<http://orientacion.sunat.gob.pe/images/imagenes/contenido/comprobantes/Manual-de-Consulta-Integrada-de-Comprobante-de-Pago-por-ServicioWEB_v2.pdf>

1. Crear un nuevo formulario con el siguiente diseño:



Deberá mostrar el código fuente de la página web cuyo Url se ingresa en el campo “Url”. Para ello deberá hacer una llamada a la Url con el verbo GET.

