|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| C:\Users\PALZATE\AppData\Local\Microsoft\Windows\INetCacheContent.Word\logo_insoft.png | | **Instructivo para realizar solicitudes REST al Agente de servicios web desde Visual Basic 6** | | | | |
| **Depto.** | Ingeniería | | **Creación** | 01/12/2017 | **Código** | RI-ING-061 |
| **Tipo doc.** | Reporte investigación | | **Modificación** |  | **Versión** | 1 |

Contenido

[Objetivo 2](#_Toc500753552)

[JSON 2](#_Toc500753553)

[Instalación de los módulos para manejo de JSON 2](#_Toc500753554)

[Módulo Dictionary 2](#_Toc500753555)

[Módulo JSONConverter 3](#_Toc500753556)

[Objetos JSON 4](#_Toc500753557)

[Arreglos JSON 4](#_Toc500753558)

[Serializar y de serializar un JSON 5](#_Toc500753559)

[Request 6](#_Toc500753560)

[Objeto WinHttp.WinHttpRequest 6](#_Toc500753561)

[Ejemplo funcional 7](#_Toc500753562)

[GetListTerceros 8](#_Toc500753563)

[Implementación de las funciones 9](#_Toc500753564)

[Resultado de la implementación 10](#_Toc500753565)

[Bibliografía 11](#_Toc500753566)

# Objetivo

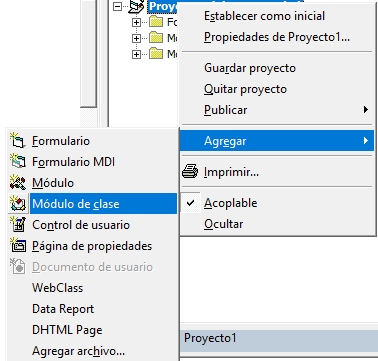
* Explicar de forma clara y precisa cómo es el manejo de JSON en el lenguaje Visual Basic 6.
* Explicar las funciones y objetos que ofrece Visual Basic 6 para realizar una petición HTTP por medio del verbo POST.
* Presentar un ejemplo funcional donde se consuma recursos de la API que ofrece el agente de servicios web de ContaPyme desde el lenguaje de programación Visual Basic 6.

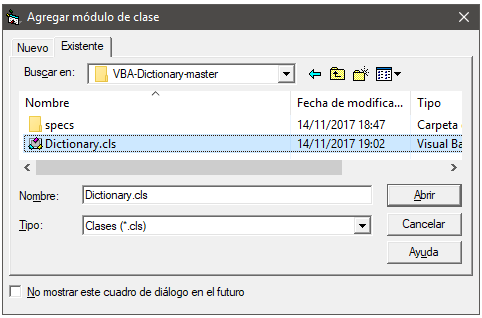
# JSON

Visual basic 6 como tal no tiene objetos nativos para el manejo de JSON pero se puede realizar este manejo mediante la librería llamada “**JsonConverter**”. Esta librería ofrece diferentes funcionalidades para el manejo de JSON a través de objetos de tipo “**Dictionary**” los cuales se pueden implementar bajo el módulo de clase llamado Dictionary.

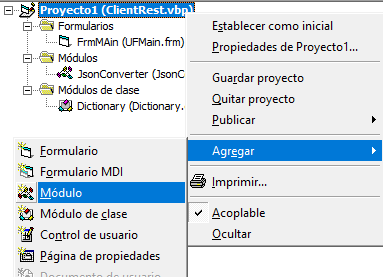
## Instalación de los módulos para manejo de JSON

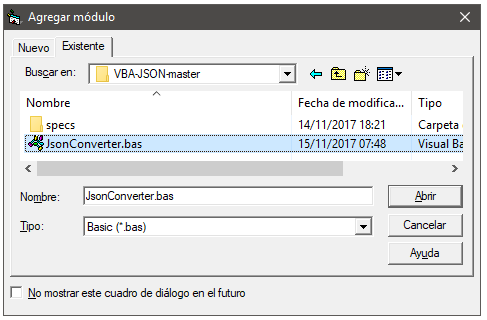
### Módulo Dictionary

Primero necesitaremos adicionar el módulo de clase para el manejo de diccionarios para el lenguaje de VB6 llamado “**VBA-Dictionary**” el cual podremos descargar desde su web oficial aquí <https://github.com/VBA-tools/VBA-Dictionary>, una vez descargado procedemos a adicionarlo a nuestro proyecto realizando los siguientes pasos:

1. Nos posicionamos sobre nuestro proyecto en Visual Basic 6, presionamos clic derecho sobre el nombre del proyecto, nos posicionamos sobre la opción de “**Agregar**” y damos clic en la opción llamada “**Módulo de clase**”.
2. En la ventana modal se nos presenta nos posicionamos sobre la pestaña llamada existente, buscamos y seleccionamos el archivo llamado “**Dictionary.cls**”, paso seguido presionamos clic sobre la opción abrir.

### Módulo JSONConverter

Para agregar este módulo es requerido haber agregado al proyecto el módulo de clase llamado “**Dictionary**” puesto que es requerido por este. Este módulo ofrecerá las funciones y objetos necesarios para el manejo de JSON, permitiendo convertir cadenas de texto para convertirlas en diccionarios y viceversa, para descargar esta librería lo podremos hacer desde la web oficial en <https://github.com/VBA-tools/VBA-JSON>, y una vez descargada ejecutar los siguientes pasos:

1. Nos posicionamos sobre nuestro proyecto en Visual Basic 6, presionamos clic derecho sobre el nombre del proyecto, nos posicionamos sobre la opción de “**Agregar**” y damos clic en la opción llamada “**Módulo**”.
2. En la ventana modal se nos presenta nos posicionamos sobre la pestaña llamada existente, buscamos y seleccionamos el archivo llamado “**JsonConverter.bas**”, paso seguido presionamos clic sobre la opción abrir.

De esta manera tendremos las librerías agregadas a nuestro proyecto en Visual Basic 6 podremos trabajar con JSON.

## Objetos JSON

El manejo de objetos JSON se realiza bajo objetos de tipo Dictionary y utilizando los métodos que este ofrece para adicionar pares (add), asignar un valor o agregarlos (item), remover una llave (remove), verificar si una llave existe (Exists) o para eliminar todos los ítems de un objeto (removAll).

|  |  |
| --- | --- |
| Código Visual Basic 6 | Representación en JSONObject |
| Dim Obj = as Dictionary  Set Obj = New Dictionary  Set Obj.Item(“name”) = “foo”  ‘Obj.add(“name”,”foo”)  Set Obj.Item(“balance”) = 1000.21  ‘Obj.add(“balance”, 1000.21)  Set Obj.Item(“italia”) = true  ‘Obj.add(“italia”, true) | {  "name": " foo ",  "num": 100,  "balance": 1000.21,  "Italia": true  } |

## Arreglos JSON

El manejo de arreglos JSON en Visual Basic 6 lo podemos hacer mediante arreglo convencionales los cuales se pueden agregar como valores a objetos previamente definidos o bajo objetos de tipo “**Collection**” según sea el caso, es decir si se necesita un JSONArray que sólo tendrá llaves de tipo String es más fácil definirlo con arreglos convencionales, pero si se necesita definir un arreglo con diferentes tipos de datos es mejor definirlo como una colección (Collection).

|  |  |
| --- | --- |
| Código Visual Basic 6 | Representación en JSONArray |
| Dim Arr as Collection  Set Arr = New Collection  Arr.add "foo"  Arr.add 100  Arr.add 1000.21  Arr.add true  Arr.add new Dictionary | [  "foo",  100,  1000.21,  True,  {}  ] |

## Serializar y de serializar un JSON

El módulo de JSONConverter también ofrece funcionalidades que permiten convertir una cadena de texto en un objeto JSON la cual será muy útil para convertir la respuesta de la petición, para esto podremos usar la propiedad ParseJson del objeto JsonConverter:

|  |
| --- |
| Código Visual Basic 6 para decodificar un JSON |
| Dim str as String = ‘{  “name”:”foo”,  “num”:100,  “balance”: 1000.21,  “Italia”:true,  “obj”:{……}  }’  Dim obj as Dictionary  Set obj = JsonConverter.ParseJson(str) |

La serialización que podemos definir como el proceso para convertir un JSON a una cadena de texto, la cual será muy útil para convertir un objeto Dictionary a un string para enviarlo en una petición la podremos realizar mediante el método ConvertToJson de la clase JsonConvert.

|  |  |
| --- | --- |
| Código Visual Basic .6 | Representación del JSON |
| Dim Obj = as Dictionary  Set Obj = New Dictionary  Obj.add(“name”,”foo”)  Obj.add(“balance”, 1000.21)  Obj.add(“italia”, true)  Obj.add(“child”, New Dictionary)  Dim Str as String  str = JsonConvert. ConvertToJson(Obj) | {  "name": " foo ",  "num": 100,  "balance": 1000.21,  "is\_vip": true,  "child":{}  } |

# Request

## Objeto WinHttp.WinHttpRequest

Esta objeto es utilizado para realizar las solicitudes de recursos a un servidor, para esto proporciona propiedades y método comunes para enviar y recibir datos de un recurso identificado por una URL o URI.

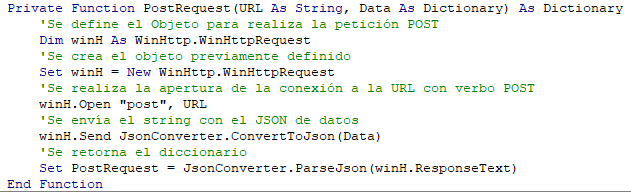
|  |
| --- |
| Código Visual Basic 6 para definir una un objeto WinHttp.WinHttpRequest y hacer una solicitud |
| --Se define la variable con el objeto  Dim winH As WinHttp.WinHttpRequest  --Se crea el Objeto para realizar la petición  Set winH = new WinHttp.WinHttpRequest  --En esta variable llegará la respuesta como un string  Dim strResult as String  --Se define la URL a donde se realizará la solicitud  Dim URL as String  URL = ‘http://exapmple.com/idrecurso’  --Se abre la conexión con el servidor definiendo el verbo  winH.Open "post", URL  --Se genera la solicitud enviando un JSONObject  winH.Send “{}”  --Por último se lee la respuesta en formato string  strResult = winH.ResponseText |

# Ejemplo funcional

El siguiente ejemplo realiza una petición que muestra el listado de terceros obtenidos de un agente de servicios web de ContaPyme (ASW) en un ListBox. Para esto primero se debe hacer GetAuth contra el agente para obtener el KeyAgent con el cual se realizará la solicitud de listado de terceros.

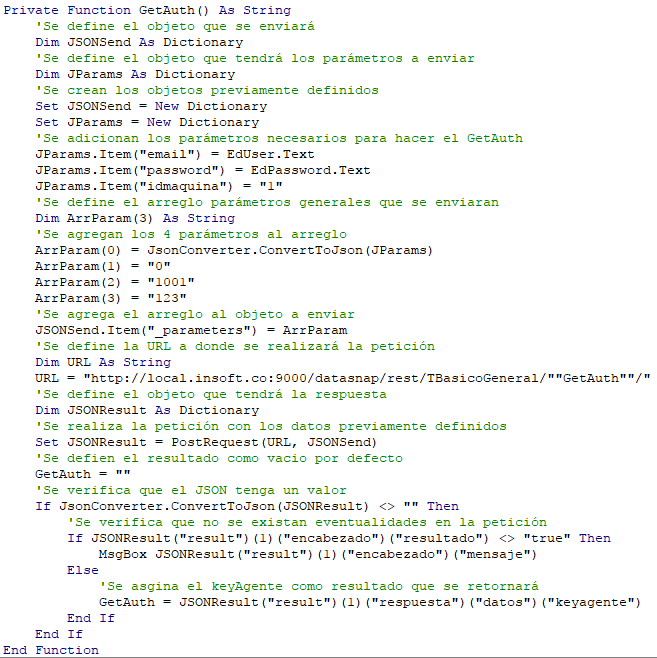
PostRequest

El objetivo de esta función es realizar una petición POST a una URL que recibe como parámetro, para esto la función también recibe un objeto de tipo Dictionary que será el que enviará en el cuerpo de la petición, la función después de hacer la solicitud retornara un objeto de tipo Dictionary que se ha entregado como respuesta.



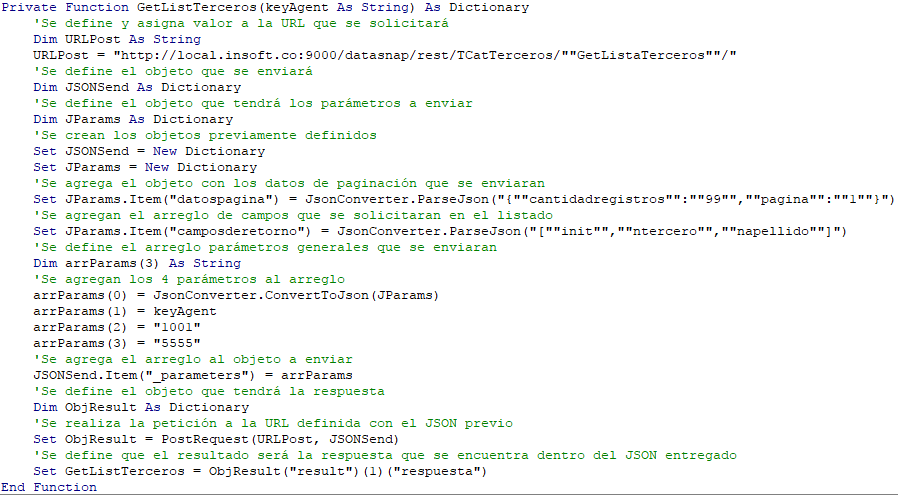
GetAuth

Esta función construirá el JSON necesario para la autenticación contra el Agente de servicios web (ASW), paso seguido realizará la solicitud de GetAuth, y retornará el KeyAgente entregado por el servidor. Para esto tomará los valores de usuario y contraseña de los dos textbox creados en diseño.



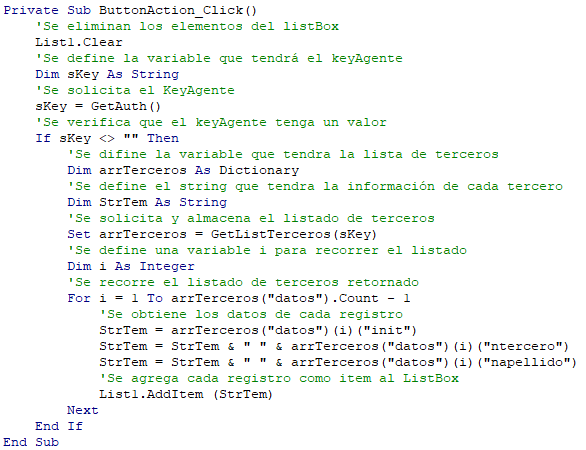
## GetListTerceros

Esta función recibe el keyAgente obtenido previamente y construye el JSON para realizar la petición de obtener el listado de terceros, como resultado esta función retornara un Dictionary con el listado de terceros.



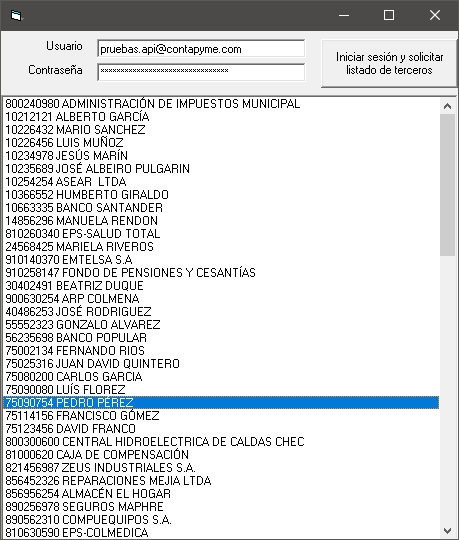
## Implementación de las funciones

La implementación se realiza sobre el evento clic del botón, en este lo primero que se hace es obtener el keyAgente y validar que este sea diferente de vacío, paso seguido se obtiene el listado de terceros el cual es almacenado en un diccionario el cual se recorrerá para obtener cada uno de los objetos que se encuentran definidos en este y agregarlos al control ListBox.



## Resultado de la implementación





# Bibliografía

Librería para manejo de diccionarios en Visual Basic 6

* <https://github.com/VBA-tools/VBA-Dictionary>

Librería recomendada para manejo de JSON

* <https://github.com/VBA-tools/VBA-JSON>

Tutorial del manejo de la clase WinHttpRequest

* <https://msdn.microsoft.com/en-us/library/windows/desktop/aa384106%28v=vs.85%29.aspx?f=255&MSPPError=-2147217396>

Video tutorial para hacer una petición post con el uso del Objeto WinHttpRequest

* <https://www.youtube.com/watch?v=NsUctonaCh4>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CONTROL DE CAMBIOS** | | | | |
| **FECHA** | **DESCRIPCIÓN** | **AUTORIZACIÓN** | **ELABORÓ** | **VERSIÓN** |
| 01/12/2017 | Se crea la versión inicial del documento. | A-13345 | PRENDON | 1 |