



RIB

Presto

API: Desarrollo de complementos

Los complementos, macros o *plugins* son programas independientes de Presto que realizan tareas específicas sobre las obras mediante una *Application Programming Interface*, API.

Copyright © 2024 by RIB Software GmbH and its subsidiaries.

This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise.

Índice

Desarrollo de complementos	3
Requisitos	3
Visual Basic Scripting VBS	3
Diferencias entre API y WebAPI	4
API.....	4
WebAPI	4
Cambios en las funciones de WebAPI respecto de API	4
Transformación de una aplicación API en WebAPI	5
Instalación WebAPI.....	6
Presto Server	6
Configurar IP y puerto	6
ASP.NET Runtime.....	7
Internet Information Services (IIS)	7
Agregar el sitio web	7
Modificar el archivo "hosts".....	9
Test de la instalación.....	10
Ejemplos	11
Recorrer 4 niveles del árbol	11
API (C#).....	11
API (VB.NET)	12
Crear una instancia a una obra en Presto Server	12
WebAPI (C#)	12
WebAPI (VB.NET).....	13
Listar las obras de un directorio de Presto Server	13
WebAPI (VB.NET).....	13

Desarrollo de complementos

Las funciones del API permiten realizar todo tipo de operaciones, creación, modificación y borrado de los registros de las obras de Presto, incluyendo el acceso a las opciones del programa.

Los usos de los complementos incluyen tareas complementarias a las realizadas directamente con Presto, como las que se suministran en la instalación, que han sido desarrollados en RIB con los mismos recursos disponibles para todos los usuarios.

Además, permiten la personalización ilimitada de Presto:

- Exportaciones e importaciones personalizadas a y desde Excel y otros programas y formatos, utilizando su propio API o mediante archivos de intercambio.
- Operaciones personalizadas sobre los datos de una obra o de un conjunto de ellas.

Estos complementos se pueden desarrollar por el mismo usuario o se pueden encargar a RIB y a sus representantes comerciales.

Los complementos suministrados con Presto se muestran en el menú dinámico "Complementos".

El código de los complementos escritos en Visual Basic Scripting, VBS, se puede consultar y modificar libremente por el usuario.

Los usuarios de licencias actualizadas pueden solicitar el código fuente de los demás complementos, realizados en VB, C# y otros lenguajes.

Requisitos

Para escribir un complemento es necesario dominar algún lenguaje de programación capaz de interactuar con un servidor de automatización y conocer la estructura interna de las tablas de Presto, tal como se aplican, por ejemplo, en el diseño de informes.

La lista de funciones figura en la nota técnica "API Funciones detalladas" con toda la información necesaria y ejemplos de uso de cada una, en Visual Basic, VB, y Visual Basic Scripting, VBS.

Visual Basic Scripting VBS

Además de los complementos suministrados con Presto en VBS, en la web de RIB puede encontrar un tutorial y un ejemplo de un complemento en VBS.

La ventaja de VBS es la sencillez de su puesta en marcha, sin necesidad de un entorno profesional de programación.

Puede escribirse el código en un editor de texto, pero se recomienda un editor especializado, como NotePad++, que tiene facilidades específicas para la escritura de código, o un entorno como Visual Studio Code.

Las desventajas de VBS son la dificultad de depurar el código, por lo que hay que recurrir a imprimir variables, y la falta de ayudas para escribir las opciones del API.

Observe que en VBS se entrega el código fuente abierto ".VBS" mientras que en los lenguajes compilados se entrega un programa ejecutable ".EXE" y la entrega del código fuente es opcional.

Si mientras desarrolla un complemento se bloquea Presto y no puede volver a iniciarlo, en el "Administrador de tareas" detenga el servicio "Presto.exe".

Diferencias entre API y WebAPI

API

El API permite interactuar con las obras ejecutando Presto localmente, por lo que es necesario disponer de una licencia de Presto en el equipo donde se ejecuta el complemento.

Es necesario registrar el componente COM ejecutando al menos una vez Presto como administrador en el equipo en el que se va a programar o ejecutar el complemento.

	Nombre	Versión
	PortableDeviceClassExtension 1.0 Type Library	1.0
	PortableDeviceConnectAPI 1.0 Type Library	1.0
	PortableDeviceTypes 1.0 Type Library	1.0
<input checked="" type="checkbox"/>	Presto 18.0 Type Library	20.0

Librería de tipos

La aplicación requiere crear un objeto de la clase "Presto.App.18". Si utiliza un compilador, agregue la referencia a este componente, que figura en "Presto 18.0 Type Library".

WebAPI

WebAPI es un Web Service que mediante llamadas REST permite interactuar con las obras accesibles en Presto Server a través de peticiones POST, pudiendo llegar a crear páginas web sobre estas obras, con autenticación mediante login y contraseña. No es necesario disponer de una licencia de Presto en los equipos dónde se va a ejecutar el complemento.

La instalación se describe en el apartado siguiente.

Cambios en las funciones de WebAPI respecto de API

- Todos los métodos que en API devolvían "void" en WebAPI devuelven "int", 0 si la operación se ha ejecutado correctamente y otro valor en caso contrario.
- La función "ReadOnly" pasa de entero a booleana.
- Las funciones "GetField" y "GetFieldBinary" pasan de dynamic a string.
- Algunos métodos pueden haber variado su retorno.

Las diferencias en las funciones se enumeran la nota técnica "API Funciones detalladas".

Transformación de una aplicación API en WebAPI

Para transformar una aplicación hay que sustituir la referencia a la librería del API "Presto 18.0 Type Library" por las referencias a las librerías "PrestoCloudJsonModel" y "PrestoCloudApiClient".

Form1	
1	Imports Presto.CloudApiClient
2	Imports PrestoCloudJsonModel

Referencia a las librerías

Y reemplazar la creación del antiguo objeto COM, como las referencias que se muestran en la tabla, por una instancia de WebAPI a una obra en Presto Server.

LENGUAJE	CONTENIDO
C#	Type oPrestoType = Type.GetTypeFromProgID("Presto.App.18") PrestoApplication prestoApplication = (PrestoApplication)Activator.CreateInstance(oPrestoType);
VB.NET	Dim Obra As PrestoLib.PrestoApplication Obra = CreateObject("Presto.App.18") Set Obra = GetObject ("", "Presto.App.18")

La forma más cómoda es utilizar un método para crear la instancia de WebAPI y luego reemplazar el antiguo objeto por el nuevo. Si utiliza el método que se incluye más adelante en los ejemplos "CreatePrestoApilnstance" tendrá que sustituir las llamadas al antiguo objeto "Obra" por "m_prestoCom".

Como el complemento ahora no tendrá acceso al interfaz de Presto hay que eliminar del código las referencias a funciones del API que dependen del interfaz: "SetModal", "SetUpdateScreen", etc.

Por último, como la comunicación con Presto Server utiliza una sesión de usuario es recomendable, después de realizar las operaciones con la obra, liberar los recursos cerrando el objeto COM y la sesión de usuario.

C#

```
m_prestoCom.Close();  
m_prestoCom = null;  
m_prestoUser.Logout();  
m_prestoUser = null;
```

VB.NET

```
m_prestoCom.Close()  
m_prestoCom = Nothing  
m_prestoUser.Logout()  
m_prestoUser = Nothing
```

Instalación WebAPI

Es necesario incluir en el proyecto las librerías "PrestoCloudJsonModel.dll", "PrestoCloudApiClient.dll" y "Newtonsoft.Json.dll" que encontrará en el directorio de instalación de Presto en la carpeta "PrestoCloudWebApi". Como estos archivos también son necesarios para la ejecución de los complementos por el usuario, deberá copiarlos en el mismo directorio donde se encuentren los complementos.

Presto Server

WebAPI requiere tener instalado Presto Server, preferiblemente en el mismo equipo dónde se vaya a ejecutar. Compruebe que Presto Server está funcionando, por ejemplo, accediendo a una de las obras desde Presto.

Presto Server debe estar activado como "PrestoServerCloud". El proceso de instalación de Presto Server se describe en la nota técnica "Instalación de Presto Server".

Entorno de trabajo						
Generales	Act	Alias	Dirección IP	Puerto	Usuario	Contraseña
PrestoServer	1	<input checked="" type="checkbox"/> PSRV	192.168.50.43	5200	Administrador	admin  ...

Entorno de trabajo: Presto Server

Configurar IP y puerto

Copie a "C:\\" la carpeta "PrestoCloudWebApi" que encontrará en el directorio de instalación de Presto.

Abra con cualquier editor de textos el archivo:

<C:\PrestoCloudWebApi\appsettings.json>

Introduzca la "Ip" y el "Puerto" donde esté instalado Presto Server, como figuran en "Entorno de trabajo: Presto Server":

```
1 {
2   "Logging": {
3     "LogLevel": {
4       "Default": "Information",
5       "Microsoft": "Warning",
6       "Microsoft.Hosting.Lifetime": "Information"
7     }
8   },
9   "PrestoServer": {
10    "Ip": "192.168.50.43",
11    "Version": "24.05",
12    "Puerto": 5200,
13    "Timeout": 60
14  },
15  "AllowedHosts": "*"
16 }
```

Configuración de "appsettings.json"

ASP.NET Runtime

Debe estar instalado "ASP.NET Core Hosting Bundle", porque son necesarios los componentes "NETCore", "AspNetCore" y "Microsoft .NET - Windows Server Hosting". Puede comprobar si están instalados "NETCore" y "AspNetCore" ejecutando desde la línea de comandos de Windows la sentencia:

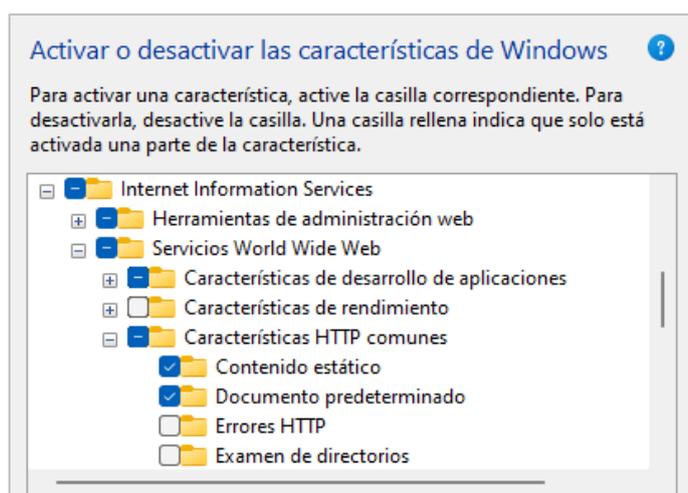
```
dotnet --list-runtimes
```

La versión instalada debe ser igual o posterior a la que figura en el archivo "PrestoCloudWebApi.runtimeconfig.json" de la carpeta "PrestoCloudWebApi".

Si está instalado "Microsoft .NET - Windows Server Hosting" aparecerá en la lista de "Aplicaciones instaladas" de Windows.

En caso de no estar instalado alguno de los componentes, puede descargar la última versión de "ASP.NET Core Hosting Bundle" desde <https://dotnet.microsoft.com>.

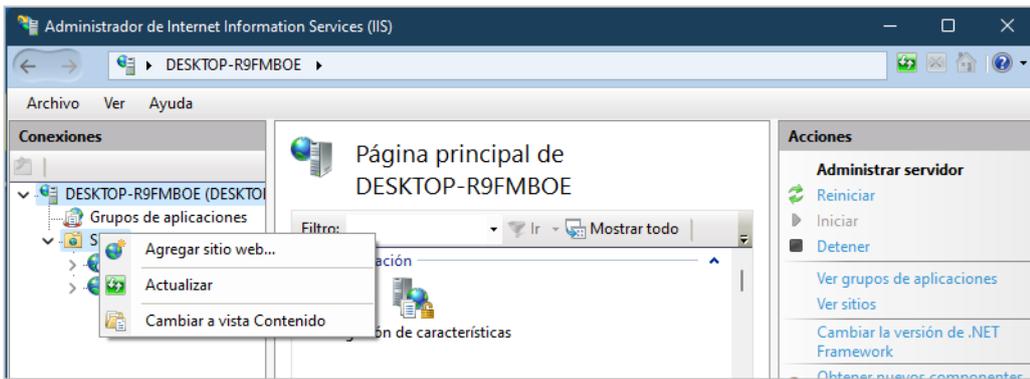
Internet Information Services (IIS)



Activar o desactivar las características de Windows

Agregar el sitio web

Acceda al administrador de Internet Information Services (IIS) y sobre "Sitios" active "Agregar sitio web...".



Rellene los valores solicitados

Agregar sitio web

Nombre del sitio: Grupo de aplicaciones:

Directorio de contenido

Ruta de acceso física:

Autenticación de paso a través

Enlace

Tipo: Dirección IP: Puerto:

Nombre de host:

Valores para añadir el sitio web

Compruebe con "Probar configuración" que los valores son válidos.

Conexión de prueba

Resultados:

Prueba	Configuración
✓ Autenticación	Nombre de usuario (RIBSPAIN\Jorge)
✓ Autorización	La ruta de acceso es accesible (C:\PrestoCloudWebApi).

Conexión de prueba

En caso de error cierre la ventana "Conexión de prueba" y pulse en "Conectar como". Seleccione "Usuario específico" y añada el usuario, con su dominio, y la contraseña con la que se ha accedido al equipo.

Si lo desea puede seleccionar que el acceso sea a través de un enlace "https".

Enlace

Tipo: **https** Dirección IP: **192.168.50.43** Puerto: **443**

Nombre de host:

Requerir indicación del nombre de servidor

Deshabilitar TLS 1.3 a través de TCP Deshabilitar QUIC

Deshabilitar TLS heredada Deshabilitar HTTP/2

Deshabilitar asociación de OCSP

Certificado SSL: **Seleccione uno**

IIS Express Development Certificate

Configuración de enlace "https"

Seleccione un certificado SSL válido.

Pruebe la configuración para comprobar que las credenciales son válidas.

Compruebe que el sitio se ha añadido correctamente.

Nombre	Id.	Estado	Enlace	Ruta de acceso
Default Web Site	1	Iniciado (http)	*:80 (http)	%SystemDrive%\inetpub\wwwroot
webapi.presto.es	2	Iniciado (http)	webapi.presto.es on 192.168.50.43:80 (http)	C:\PrestoCloudWebApi
webapi2.presto.es	3	Iniciado (http)	webapi.presto.es on 192.168.50.43:443 (https)	C:\PrestoCloudWebApi

Sitios web agregados en IIS

Modificar el archivo "hosts"

Abra el archivo "C:\windows\system32\drivers\etc\hosts" con un editor de textos como administrador y añada una línea con la dirección IP de la máquina y el nombre del host "webapi.presto.es", separados por un tabulador.

```
# localhost name resolution is handled within DNS itself.  
# 127.0.0.1 localhost  
# ::1 localhost  
192.168.50.43 webapi.presto.es
```

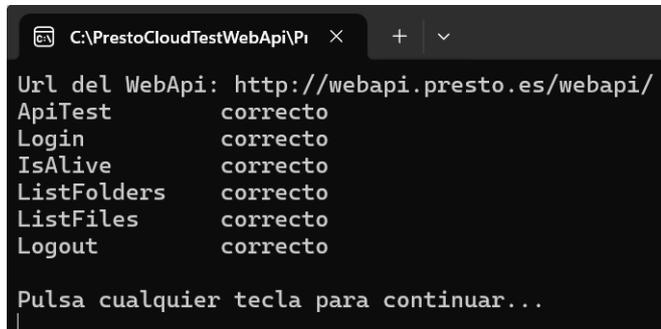
Dirección IP y nombre del host

Opcionalmente puede introducir como nombre del host "iplocal".

Las llamadas al servidor desde el WebAPI se deben hacer utilizando el nombre DNS, no la dirección IP.

Test de la instalación

Para comprobar la configuración de WebAPI ejecute la aplicación [PrestoCloudTestWebApi.exe](#).



```
C:\PrestoCloudTestWebApi\Pi x + v
Url del WebApi: http://webapi.presto.es/webapi/
ApiTest      correcto
Login        correcto
IsAlive      correcto
ListFolders  correcto
ListFiles    correcto
Logout       correcto

Pulsa cualquier tecla para continuar...
```

Resultado

Si ApiTest da correcto, y los demás elementos incorrecto, quiere decir que hay comunicación con el WebAPI, pero no entre el WebAPI y PrestoServer.

Si el test indica que hay diferente versión entre PrestoServer y el WebAPI, en el IIS, “Administrar sitio web”, haga clic en “Detener”, para poder copiar los archivos de la versión actual.

Copie ahora los archivos de la carpeta “C:\Program Files\Presto 2024.0x\PrestoCloud”:

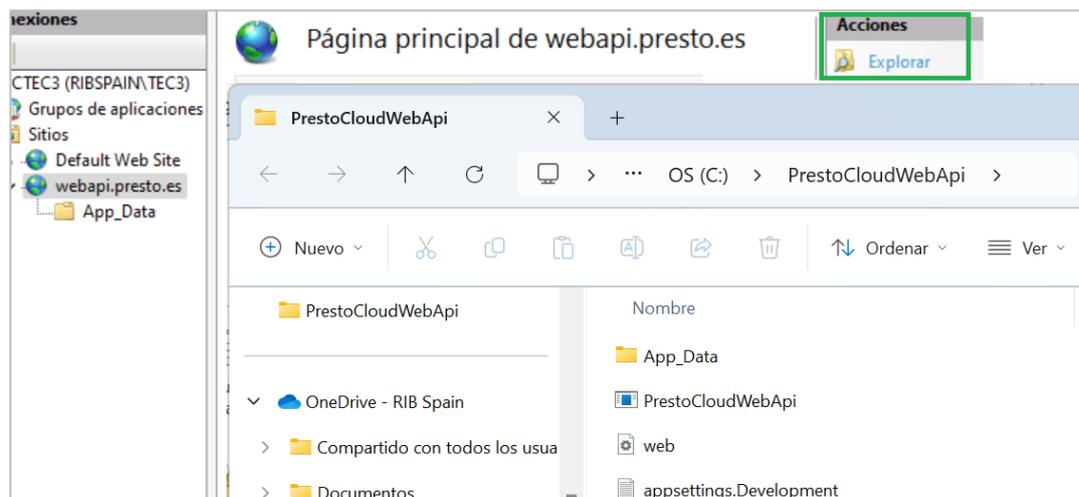
[PrestoCloudApiClient](#)

[PrestoCloudTestWebApi](#)

a las carpetas C:\PrestoCloudWebApi y C:\PrestoCloudTestWebApi, respectivamente.

La versión debe coincidir a nivel del segundo dígito, por ejemplo 24.05.

En el IIS, clique “Acciones: Explorar” y compruebe que los archivos que aparecen son los que corresponden a los archivos que acaba de copiar de la carpeta de instalación de Presto.



Directorio

Revise “appsettings.json”

```

"PrestoServer": {
  "Ip": "192.168.50.43",
  "Version": "24.05",
  "Puerto": 5200,
  "Timeout": 60
},

```

Y en el IIS clique "Iniciar"

Ejemplos

Recorrer 4 niveles del árbol

API (C#)

```

Private void SurroundingSub ()
{
  Object obra;
  String code;
  String name;
  String superior;
  var CodeInf;
  obra = Interaction.GetObject ("", "Presto.App.18");
  superior = "###";
  obra.SetElement (1, "Relaciones", "Relaciones.CodSup", Strings.Chr (34) + superior +
Strings.Chr (34), "Conceptos.Nat!=219");
  While (obra.GetElement (1) == 0)
  {
    superior = obra.GetFieldStr ("Relaciones.CodInf");
    code = obra.GetFieldStr ("Conceptos.Código");
    name = obra.GetFieldStr ("Conceptos.Resumen");
    obra.SetElement (2, "Relaciones", "Relaciones.CodSup", Strings.Chr (34) + superior +
Strings.Chr (34), "Conceptos.Nat!=219");
    While (obra.GetElement (2) == 0)
    {
      superior = obra.GetFieldStr ("Relaciones.CodInf");
      code = obra.GetFieldStr ("Conceptos.Código");
      name = obra.GetFieldStr ("Conceptos.Resumen");
      obra.SetElement (3, "Relaciones", "Relaciones.CodSup", Strings.Chr (34) + superior +
Strings.Chr (34), "Conceptos.Nat!=219");
      While (obra.GetElement (3) == 0)
      {
        superior = obra.GetFieldStr ("Relaciones.CodInf");
        code = obra.GetFieldStr ("Conceptos.Código");
        name = obra.GetFieldStr ("Conceptos.Resumen");
        obra.SetElement (4, "Relaciones", "Relaciones.CodSup", Strings.Chr (34) + superior +
Strings.Chr (34), "Conceptos.Nat!=219");
        While (obra.GetElement (4) == 0)
        {
          superior = obra.GetFieldStr ("Relaciones.CodInf");
          code = obra.GetFieldStr ("Conceptos.Código");
          name = obra.GetFieldStr ("Conceptos.Resumen");

```

```

    }
  }
}

```

API (VB.NET)

```

Private Sub SurroundingSub ()
    Dim obra As Object
    Dim code As String
    Dim name As String
    Dim superior As String
    obra = GetObject ("", "Presto.App.18")
    superior = "###"
    obra.SetElement (1, "Relaciones", "Relaciones.CodSup", Chr (34) & superior & Chr (34),
"Conceptos.Nat!=219")
    While obra.GetElement (1) = 0
        superior = obra.GetFieldStr ("Relaciones.CodInf")
        code = obra.GetFieldStr ("Conceptos.Código")
        name = obra.GetFieldStr ("Conceptos.Resumen")
        obra.SetElement (2, "Relaciones", "Relaciones.CodSup", Chr (34) & superior & Chr (34),
"Conceptos.Nat!=219")
        While obra.GetElement (2) = 0
            superior = obra.GetFieldStr ("Relaciones.CodInf")
            code = obra.GetFieldStr ("Conceptos.Código")
            name = obra.GetFieldStr ("Conceptos.Resumen")
            obra.SetElement (3, "Relaciones", "Relaciones.CodSup", Chr (34) & superior & Chr (34),
"Conceptos.Nat!=219")
            While obra.GetElement (3) = 0
                superior = obra.GetFieldStr ("Relaciones.CodInf")
                code = obra.GetFieldStr ("Conceptos.Código")
                name = obra.GetFieldStr ("Conceptos.Resumen")
                obra.SetElement (4, "Relaciones", "Relaciones.CodSup", Chr (34) & superior & Chr
(34), "Conceptos.Nat!=219")
                While obra.GetElement (4) = 0
                    superior = obra.GetFieldStr ("Relaciones.CodInf")
                    code = obra.GetFieldStr ("Conceptos.Código")
                    name = obra.GetFieldStr ("Conceptos.Resumen")
                End While
            End While
        End While
    End While
End Sub

```

Crear una instancia a una obra en Presto Server

WebAPI (C#)

```

private static PrestoClientApiUser m_prestoUser = null;
private static PrestoClientApiCom m_prestoCom = null;
private static void CreatePrestoApiInstance()
{
    if (m_prestoUser == null)
    {
        m_prestoUser = new PrestoClientApiUser("http://webapi.presto.es/webapi/");
        ResponseValue returnVal = m_prestoUser.Login("Administrador", "admin");
    }
}

```

```

    if (returnValue == ResponseValue.Ok)
    {
        m_prestoCom = new PrestoClientApiCom(m_prestoUser);
        int retValue = m_prestoCom.Open("Presupuesto.Presto");
        // Ruta + obra dentro de Presto Server
        if (retValue != 0) m_prestoCom = null;
    }
}
}
}

```

WebAPI (VB.NET)

```
Public Class Form1
```

```
    Private m_prestoUser As PrestoClientApiUser = Nothing
```

```
    Private m_prestoCom As PrestoClientApiCom = Nothing
```

```
    Private Sub CreatePrestoApiInstance()
```

```
        If m_prestoUser Is Nothing Then
```

```
            m_prestoUser = New PrestoClientApiUser("http://webapi.presto.es/webapi/")
```

```
            Dim returnValue As ResponseValue = m_prestoUser.Login("Administrador", "admin")
```

```
            If returnValue = ResponseValue.Ok Then
```

```
                m_prestoCom = New PrestoClientApiCom(m_prestoUser)
```

```
                Dim retValue As Integer = m_prestoCom.Open("Presupuesto.Presto")
```

```
                '// Ruta + obra dentro de Presto Server
```

```
                If retValue <> 0 Then m_prestoCom = Nothing
```

```
            End If
```

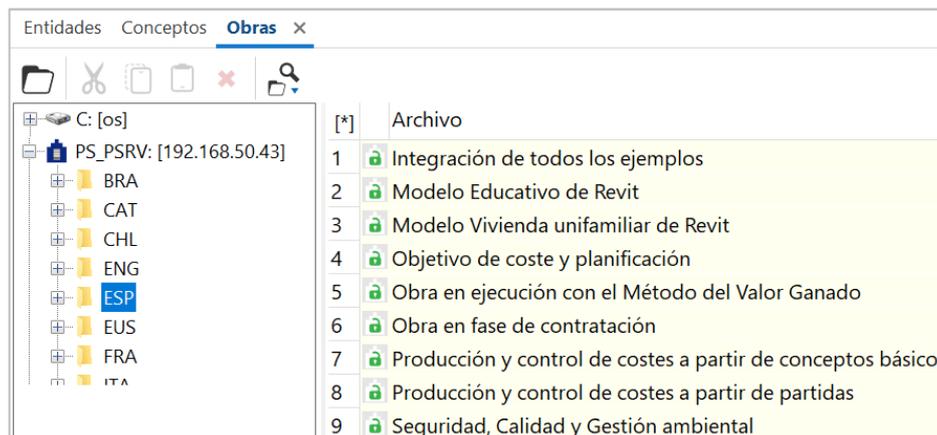
```
        End If
```

```
    End Sub
```

```
End Class
```

Listar las obras de un directorio de Presto Server

WebAPI (VB.NET)



Obras de Presto Server

Iniciación

- Crear un proyecto VB.NET (.NET Framework Versión 4.7.2 o superior, o .NET Core 2.0 o superior)
- Añadir referencia a las DLL "PrestoCloudJsonModel" y "PrestoCloudWebApi"

Reference Name	Type	Version	Copy Local	Path
PrestoCloudApiClient	.NET	1.0.0.0	True	C:\Program Files (x86)\Presto 2021.02\PrestoCloud\PrestoCloudTestWebApi\PrestoCloudApiClient.dll
PrestoCloudJsonModel	.NET	1.0.0.0	True	C:\Program Files (x86)\Presto 2021.02\PrestoCloud\PrestoCloudApiClient\PrestoCloudJsonModel.dll

Configuración del compilador

Pasos del código

- Crear un formulario con un "DataGridView1" de tres columnas, para mostrar el nombre, el importe del presupuesto y el importe del objetivo de las obras listadas.
- Crear un objeto de tipo "PrestoClientApiUser", pasando la dirección del WebAPI.
- Realizar un "Login" con un usuario válido.
- Crear un objeto de tipo "PrestoClientApiFiles".
- Realizar una llamada para obtener el listado de obras "ListFiles".
- Realizar un "Logout".

Código

```
Imports Presto.CloudApiClient
Imports PrestoCloudJsonModel
Public Class Form1
    Private m_prestoUser As PrestoClientApiUser = Nothing
    Private m_prestoCom As PrestoClientApiCom = Nothing
    Dim DirectorioPS As String = "ESP"
    Dim strUser As String = "Administrador"
    Dim strPassword As String = "admin"
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Puebla_DatagridView_Obras_PS()
    End Sub
    Sub Puebla_DatagridView_Obras_PS()
        Dim User As New PrestoClientApiUser("http://webapi.presto.es/webapi/")
        Dim responseValue As ResponseValue = User.Login(strUser, strPassword)
        If (responseValue = responseValue.Ok) Then
            Dim Files As New PrestoClientApiFiles(User)
            Dim listFiles As New List(Of PrestoCloudJsonModel.FileItemInfo)
            responseValue = Files.ListFiles(listFiles, DirectorioPS, False)
            For f = 0 To listFiles.Count - 1
                DataGridView1.Rows.Add(listFiles.Item(f).name)
            Next
        End If
        CreatePrestoApiInstance()
        DataGridView1.AllowUserToAddRows = False
        DataGridView1.AutoSizeColumns()
        User.Logout()
    End Sub
    Private Sub CreatePrestoApiInstance()
        If m_prestoUser Is Nothing Then
            m_prestoUser = New PrestoClientApiUser("http://webapi.presto.es/webapi/")
            Dim returnValue As ResponseValue = m_prestoUser.Login(strUser, strPassword)
            If returnValue = ResponseValue.Ok Then
                m_prestoCom = New PrestoClientApiCom(m_prestoUser)
                For f = 0 To DataGridView1.Rows.Count - 2
```

```
        '// Ruta + proyecto dentro de Presto Server
        Dim retVal As Integer = m_prestoCom.Open(DirectorioPS & "\" &
DataGridView1.Rows(f).Cells(0).Value)
        If retVal <> 0 Then m_prestoCom = Nothing
        With m_prestoCom
            If .FindEqual("Conceptos", "Conceptos.Nat", 0) = 0 Then
                DataGridView1.Rows(f).Cells(1).Value = .EvalNum("Conceptos.Pres")
                DataGridView1.Rows(f).Cells(2).Value = .EvalNum("Conceptos.Obj")
            End If
        End With
    Next
End If
End If
End Sub
End Class
```