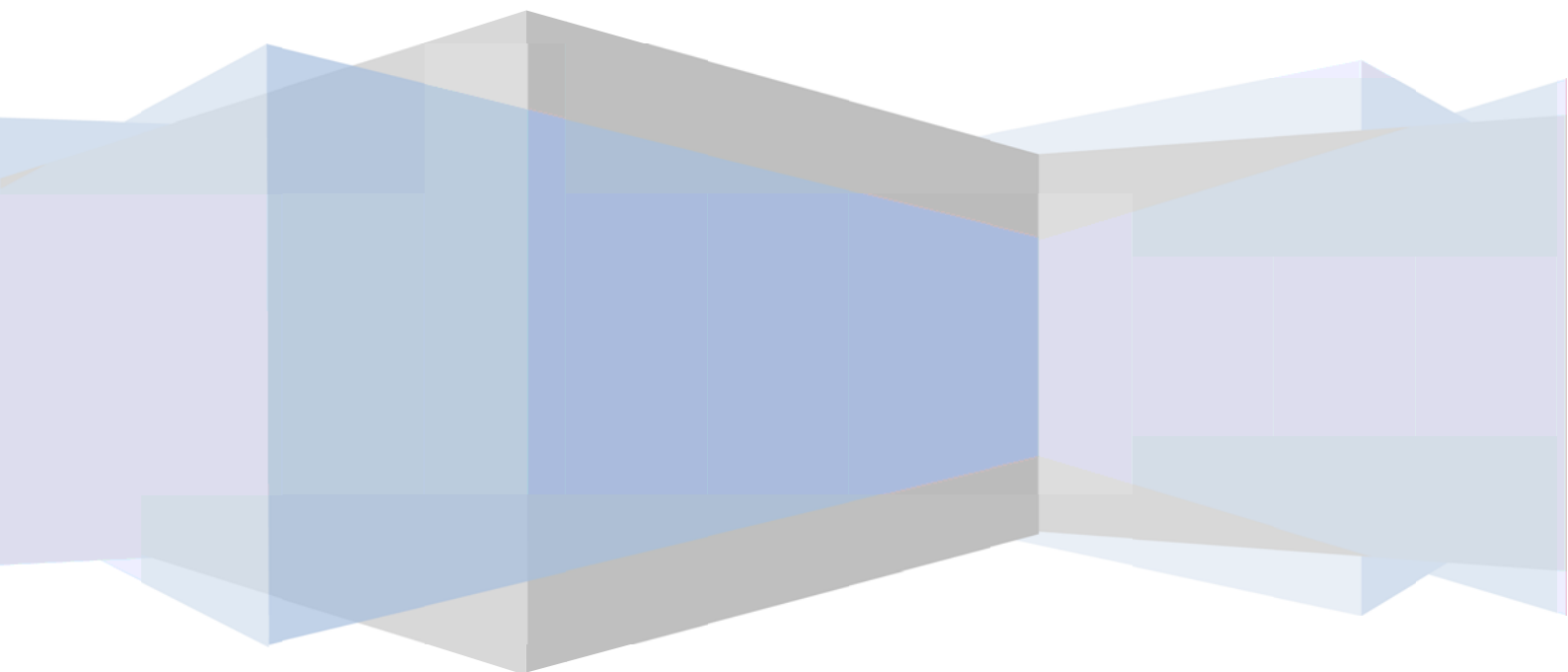


Manual de Integración de Giltza Desktop en .NET

07/05/2018



Contenido

| | |
|---|----------|
| 1. INTRODUCCIÓN | 3 |
| 2. DESCRIPCIÓN DE LA SOLUCIÓN DE EJEMPLO | 3 |
| Proyecto GiltzaDesktopSdk | 3 |
| Proyecto GiltzaDesktopSample | 6 |
| 3. ARCHIVOS COMPILADOS..... | 8 |

1. Introducción

Giltza Desktop es una forma de implementar la autenticación de Giltza para aplicaciones de escritorio. Para realizar este propósito, se proporciona una librería de ejemplo, y una aplicación que consume esa librería, que muestra cómo se puede hacer esta integración.

2. Descripción de la solución de Ejemplo

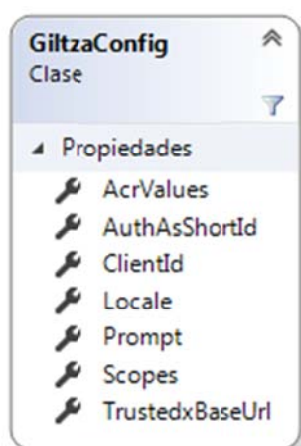
La solución que se proporciona de ejemplo consta de dos proyectos que son:

- GiltzaDesktopSdk: Este proyecto es la librería encargada de abstraer el proceso de autenticación de GiltzaDesktop.
- GiltzaDesktopSample: Este proyecto, simula ser un sistema que integra la librería que se encarga de las funciones de autenticación en Giltza desde escritorio.

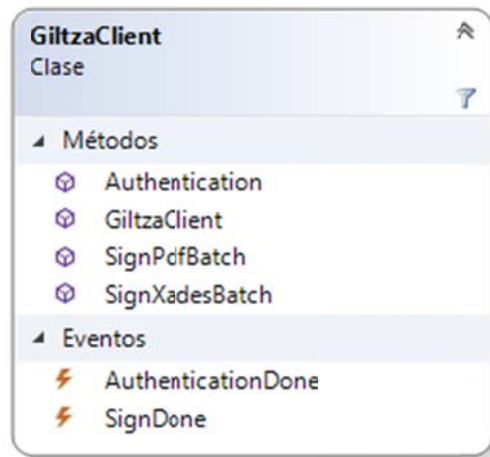
Proyecto GiltzaDesktopSdk

Este proyecto es del tipo “Biblioteca de clases”, y usa la tecnología WPF (Windows Presentation Foundation). Esta tecnología es la que nos permite mostrar una ventana con un navegador embebido, que es el que nos permitirá interactuar con Giltza.

Esta librería proporciona también una clase, que es la clase GiltzaConfig, que contiene en forma de propiedades, todas las variables que hay que pasarle a la librería para su correcto funcionamiento. Se ha evitado que sea la librería la que lea estas variables, para que pueda ser invocada desde distintos clientes, y que estos pudiesen tener cada uno sus propias variables. De esta forma las variables pasan a ser responsabilidad de la aplicación llamante. El diagrama de esta clase es:



La otra clase importante es la clase GiltzaClient, que es la encargada de interactuar con Giltza. El diagrama es el siguiente:

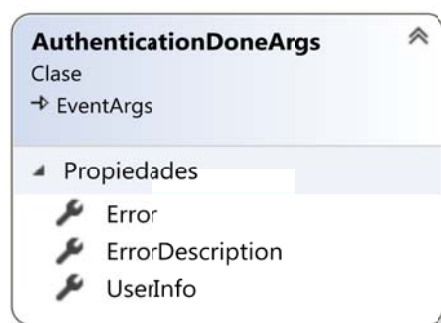


Esta clase permite realizar la operación de autenticación y la operación de firma tanto PADES como XADES.

Expone un constructor que acepta un objeto del tipo GiltzaConfig.

Una vez inicializado el objeto, se puede llamar al método “Authentication”, que es un método sin parámetros y que no devuelve nada. Este método es el encargado de iniciar el proceso de autenticación.

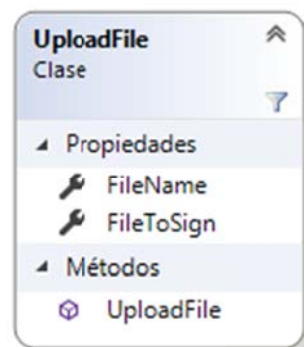
Cuando se termine el proceso de autenticación, la librería lanza un evento que debe de ser capturado por la aplicación invocante. Este evento es el evento “AuthenticationDone”, y devuelve el objeto “AuthenticationDoneArgs”, que contendrá la respuesta del proceso de autenticación en el caso de que concluya satisfactoriamente, o el error y su descripción en el caso de que algo falle. Se ha mantenido un solo evento para todo, para simplificar el proceso, pero se podrían programar tantos eventos como fuesen necesarios en función de las necesidades del integrador. El diagrama de clase de “AuthenticationDoneArgs” es el siguiente:



La propiedad “UserInfo” devuelve un String en formato JSON con toda la información del usuario autenticado, en función de la propiedad “Scopes” fijada en la clase “GiltzaConfig”. Para más información sobre los “Scopes”, consultar manual de Giltza.

Una vez realizada la autenticación, podremos proceder a realizar la firma de documentos. **Si previamente a la firma de documentos, no se ha llamado al método de autenticación, el método de firma devolverá una excepción.**

El método de firma acepta una lista de objetos del tipo UploadFile, y el fichero JSON que define el tipo de firma que queremos realizar. La clase UploadFile tiene el siguiente diagrama:



Esta clase tiene una propiedad donde indicaremos el nombre del fichero a firmar, y otra propiedad donde indicaremos el contenido del fichero a firmar en Array de Bytes.

En la lista de objetos podemos indicar tantos documentos como queramos siempre que se realice el mismo tipo de firma sobre todos los elementos de la lista.

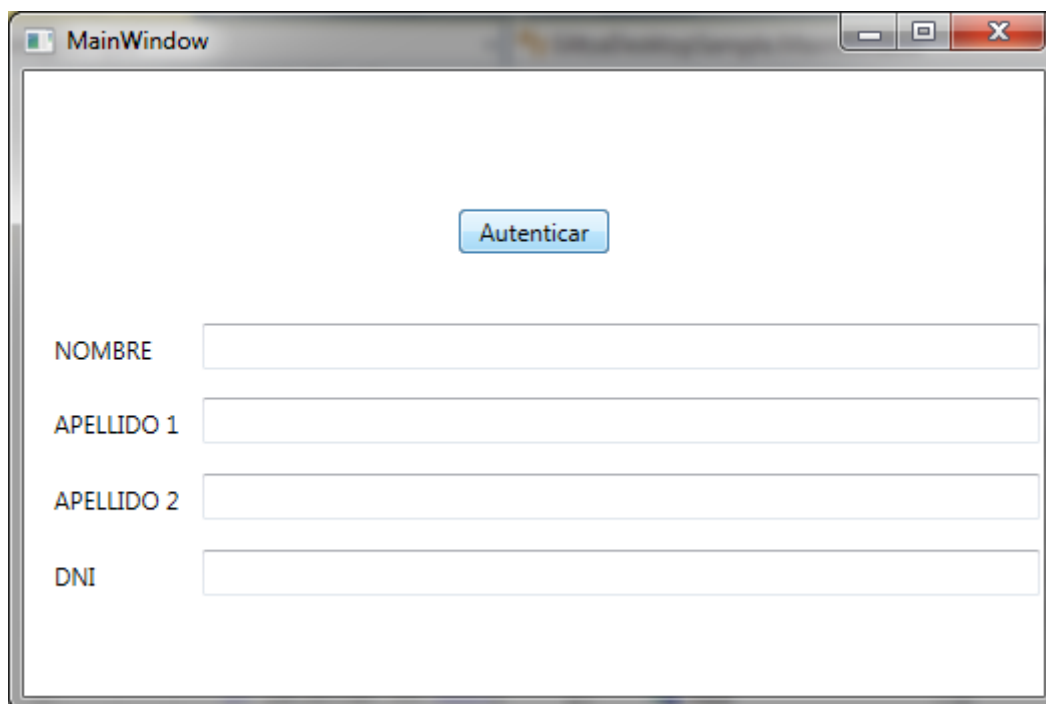
Cuando se termine el proceso de firma la librería lanza un evento que debe de ser capturado por la aplicación invocante. Este evento es el evento “SignDone”, y devuelve el objeto “SignDoneArgs”, que contendrá una lista de Array de bytes que representa los archivos firmados.



Para simplificar el tratamiento de los ficheros JSON, la librería se apoya en otra librería de terceros que es “Newtonsoft.Json.dll”

Proyecto GiltzaDesktopSample

Este proyecto sirve como ejemplo de aplicación invocante que usa la librería. Consta de una ventana que contiene un botón que desencadena la autenticación. Una vez acabada la autenticación, se rellenan los datos del usuario autenticado en la ventana.



The image shows a screenshot of a Windows application window titled "MainWindow". The window has a standard Windows title bar with minimize, maximize, and close buttons. Inside the window, there is a blue button labeled "Autenticar" centered at the top. Below the button, there are four text input fields arranged vertically. The labels for these fields are "NOMBRE", "APELLIDO 1", "APELLIDO 2", and "DNI", positioned to the left of each respective input field.

Al pinchar en el botón autenticar, se ejecuta el siguiente código:

```
private void Button_Click(object sender, RoutedEventArgs e)
{
    GiltzaConfig giltzaConfig = new GiltzaConfig();
    giltzaConfig.AuthAsShortId = Settings.Default.AuthAsShortId;
    giltzaConfig.TrustedxBaseUrl = Settings.Default.TrustedxBaseUrl;
    giltzaConfig.ClientId = Settings.Default.ClientId;
    giltzaConfig.Scopes = Settings.Default.Scopes;
    giltzaConfig.AcrValues = Settings.Default.AcrValues;

    GiltzaClient giltzaClient = new GiltzaClient(giltzaConfig);
    giltzaClient.AuthenticationDone += GiltzaClient_AuthenticationDone;
    giltzaClient.Authentication();
}

private void GiltzaClient_AuthenticationDone(object sender,
AuthenticationDoneArgs e)
{
    if (e.Error == null)
    {
        var jsonVal = JObject.Parse(e.UserInfo);
        NombreTextBox.Text = jsonVal["given_name"].ToString();
        Apellido1TextBox.Text = jsonVal["surname1"].ToString();
        Apellido2TextBox.Text = jsonVal["surname2"].ToString();
        DniTextBox.Text = jsonVal["dni"].ToString();
    } else
    {
        ErrorTextBlock.Text = e.Error + e.ErrorDescription;
    }
}
```

Se ejecuta el método “Button_Click”. En este método, primero se crea el objeto GiltzaConfig. Se leen variables del fichero de configuración, y se fijan las variables del objeto. Una vez fijadas las variables del objeto, se llama al constructor pasándole el objeto de configuración. Después asociamos el método GiltzaClient_AuthenticationDone al evento AuthenticationDone, y llamamos al método Authentication. Cuando se acabe la autenticación se ejecutará el método que hemos asociado. Una vez autenticados, podremos firmar los documentos.

```

        private void FirmarPdf_Click(object sender, RoutedEventArgs e)
        {
            ...

            List<UploadFile> filesToSign = new List<UploadFile>();
            filesToSign.Add(new UploadFile("uno.pdf",
            File.ReadAllBytes(filename)));
            filesToSign.Add(new UploadFile("dos.pdf",
            File.ReadAllBytes(filename)));
            String signatureJson = File.ReadAllText("pdfSignature.json");
            giltzaClient.SignDone += GiltzaClientPdf_SignDone;
            giltzaClient.SignPdfBatch(filesToSign, signatureJson);
        }

        private void GiltzaClientPdf_SignDone(object sender, SignDoneArgs e)
        {
            int i = 1;
            foreach (byte[] archivo in e.FicherosFirmados)
            {
                File.WriteAllBytes("sample_signed" + i + ".pdf", archivo);
                i++;
            }
            MessageBox.Show("Fichero firmado.");
        }
    }

```

3. Archivos compilados

La aplicación consta de cinco archivos, que se detallan a continuación:

- GiltzaDesktopSample.exe: Este ejecutable representa una aplicación de ejemplo que va a consumir la librería.
- GiltzaDesktopSample.exe.config: Este es el fichero de configuración que contendrá todas las variables que necesita Giltza para su correcto funcionamiento.
- GiltzaDesktopSdk.dll: Librería que se ocupa de las comunicaciones con Giltza.
- Newtonsoft.Json.dll y Newtonsoft.Json.xml: Son librerías de terceros para el parseo de ficheros JSON.