Fm.EHF.EHFUtility

Description

EHFUtility is a .NET-library to simplify EHF-related tasks in .NET. Use this utility to:

- Validate EHF-documents using DIFIs validationservice
- Send EHF-documents
- Perform SMP and SML lookup

What you can't do with this library:

- Receive PEPPOL-documents (Oxalis is a complete access point wich can receive documents and send documents (java or command-line))
- Generate EHF-xml. If you want to generate .NET-classes for EHF-Invoice: download XSD-schemas from DIFIs VEFaValidator-tool and use Xsd2Code or other tool

EHFUtility uses a modified version of PEPPOLs START-library (upgraded from .NET 3.5 to 4.5 with integrated support for claims) to send PEPPOL documents. Validation is performed using DIFIs validationservice (you can install your own, see here).

To use the library:

```
var xml = new XmlDocument();
xml.Load(@"SampleData\fminvoice.xml");
var ehfUtility = new EHFUtility();
```

```
//To validate document using DIFIs validationservice:
ValidationResult validationResult = ehfUtility.ValidateDocument(xml);
```

```
//Lookup service metadata:
SmpInformation lookupFm = ehfUtility.LookupSMP("9908:974763907");
```

```
//Send invoice:
SendResult sendResult = ehfUtility.SendDocument(xml, "9908:453463465", "9908:974763907");
```

Installation

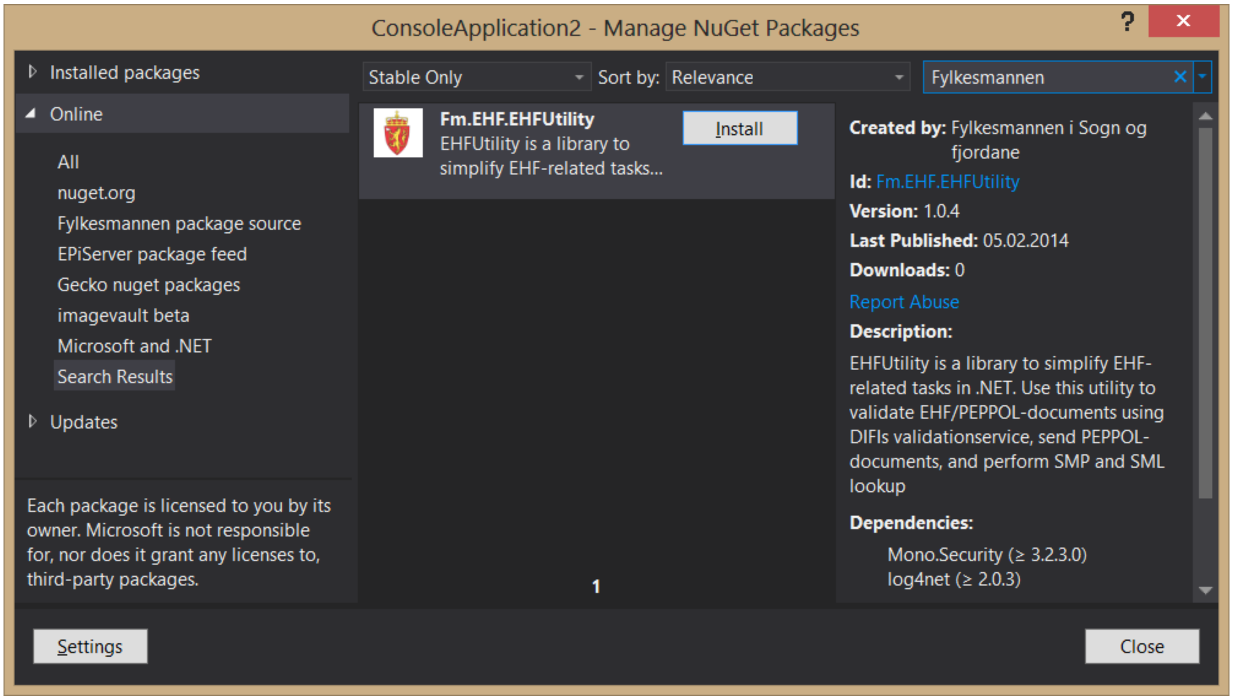
Short version:

1. Add NuGet-package
2. Install PEPPOL root certificates into trusted root certification authorities
3. Create keystore with your private key and certificate issued by PEPPOL
4. Modify app.config/web.config

Long version:

Add NuGet-package

Right click project/solution->Manage NuGet-packages->nuget.org->search for «EHF» or «Fylkesmannen»



Fylkesmannen

Fm.EHF.EHFUtility
EHFUtility is a library to simplify EHF-related tasks...

Created by: Fylkesmannen i Sogn og fjordane
Id: Fm.EHF.EHFUtility
Version: 1.0.4
Last Published: 05.02.2014
Downloads: 0
Report Abuse

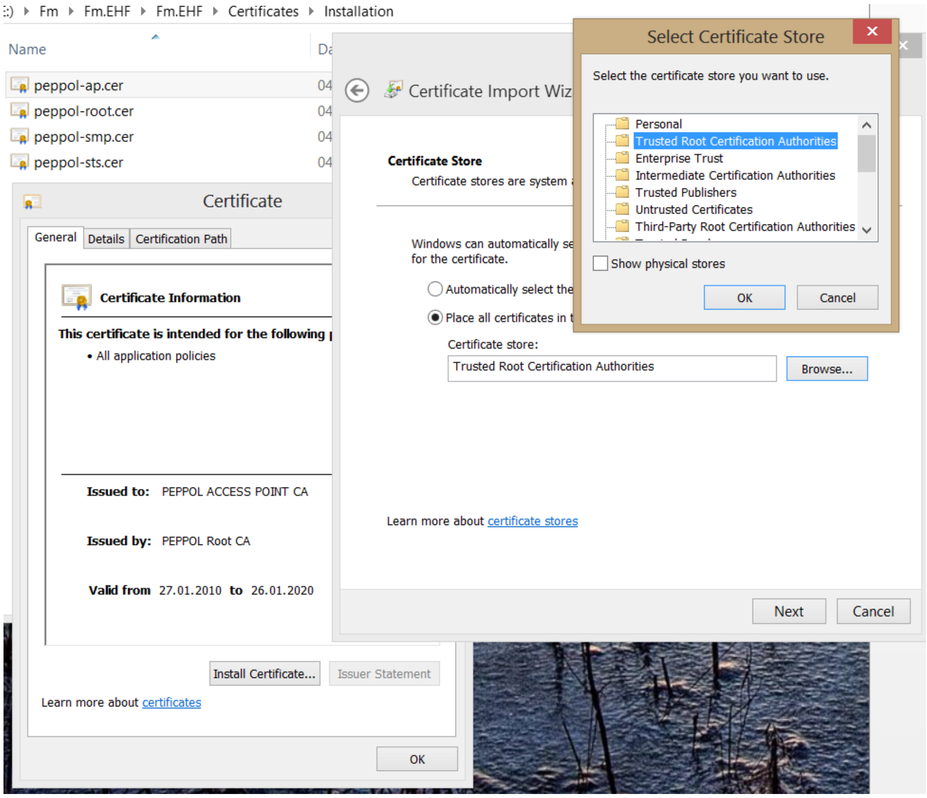
Description:
EHFUtility is a library to simplify EHF-related tasks in .NET. Use this utility to validate EHF/PEPPOL-documents using DIFIs validationservice, send PEPPOL-documents, and perform SMP and SML lookup

Dependencies:
Mono.Security (≥ 3.2.3.0)
log4net (≥ 2.0.3)

Each package is licensed to you by its owner. Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.

Install PEPPOL root certificates

The NuGet-package added a folder in your project, open folder Certificates\Installation\ in explorer. Install all root certificates from PEPPOL to Local machine->Trusted Root Certification Authorities



peppol-ap.cer 04
peppol-root.cer 04
peppol-smp.cer 04
peppol-sts.cer 04

Certificate Information

This certificate is intended for the following
- All application policies

Issued to: PEPPOL ACCESS POINT CA

Issued by: PEPPOL Root CA

Valid from 27.01.2010 **to** 26.01.2020

Learn more about certificates

Certificate Store
Certificate stores are system

Windows can automatically se
for the certificate.

○ Automatically select the
● Place all certificates in t

Certificate store:
Trusted Root Certification Authorities

Learn more about certificate stores

Select the certificate store you want to use.

☐ Show physical stores

These certificates are needed for validation of certificates from other access points, SMP, etc.

The certificates included in the NuGet-package are «live»-certificates. If you want to setup an test-environment, download certificates from PEPPOL

Download root certificates from PEPPOL here:

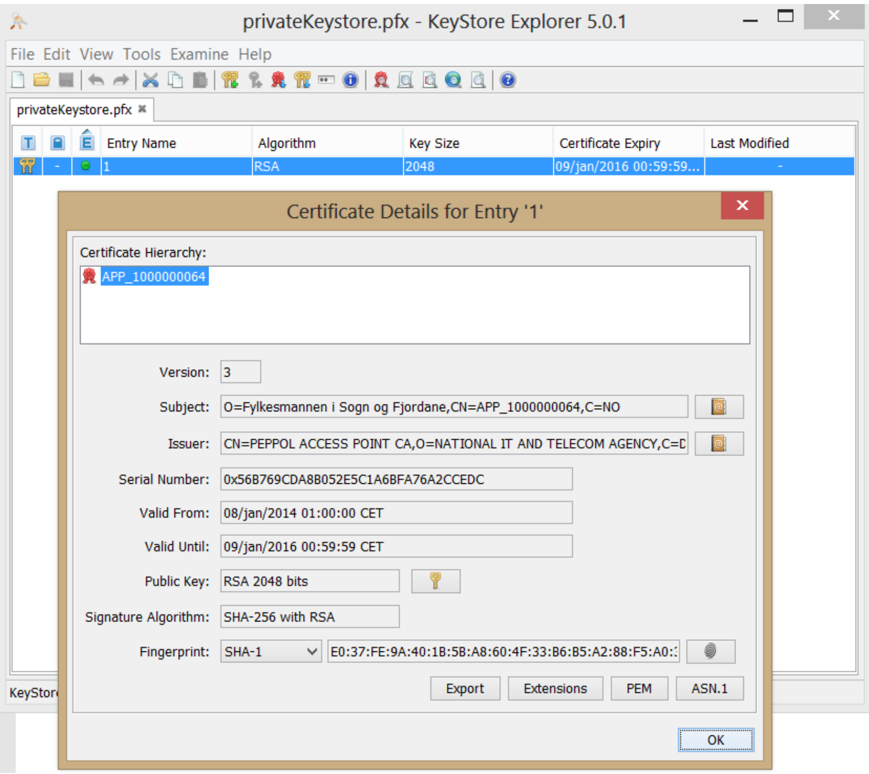
- https://onsite.verisign.com/services/DigitaliseringsstyrelsenOpenPEPPOLSECURITYTOKENSERVICECA/digitalidCenter.htm
- https://onsite.verisign.com/services/DigitaliseringsstyrelsenOpenPEPPOLACCESSPOINTCA/digitalidCenter.htm
- https://onsite.verisign.com/services/DigitaliseringsstyrelsenOpenPEPPOLSERVICEMETADATAPUBLISHERCA/digitalidCenter.htm

Install your own certificate

You need to obtain a certificate from PEPPOL identifying you to other PEPPOL-services. Look here (XXX) for more information.

1. Download KeyStore Explorer from http://keystore-explorer.sourceforge.net/downloads.php.
2. Create new keystore->PKCS #12->Save as «Certificates\privateKeystore.pfx». Enter a password «changeit» (or something else)
3. Generate your private key and certificate signing request:
   a. Right click->generate Key Pair->RSA 2048->fill in information
   b. Right click key pair->generate CSR (certificate signing request)
   c. Upload request.CSR to PEPPOL, get your certificate in return «reply from peppol.p7r»
   d. Right click key pair->Import CA reply
4. Your certificate from PEPPOL should look like this:

5. Your privateKeystore.pfx should look like this:


privateKeystore.pfx - KeyStore Explorer 5.0.1
File Edit View Tools Examine Help
privateKeystore.pfx
KeyStor

T		E	Entry Name	Algorithm	Key Size	Certificate Expiry	Last Modified
	-	●	1	RSA	2048	09/jan/2016 00:59:59...	-

Certificate Details for Entry '1'

Certificate Hierarchy:
APP_1000000064

Version: 3

Subject: O=Fylkesmannen i Sogn og Fjordane,CN=APP_1000000064,C=NO

Issuer: CN=PEPPOL ACCESS POINT CA,O=NATIONAL IT AND TELECOM AGENCY,C=D

Serial Number: 0x56B769CDA8B052E5C1A6BFA76A2CCEDC

Valid From: 08/jan/2014 01:00:00 CET

Valid Until: 09/jan/2016 00:59:59 CET

Public Key: RSA 2048 bits

Signature Algorithm: SHA-256 with RSA

Fingerprint: SHA-1 E0:37:FE:9A:40:1B:5B:A8:60:4F:33:B6:B5:A2:88:F5:A0::

It should only contain one entry! If you have two entries you did it wrong ☺

6. Configure password to your keystore in app.config:

```
<clientCertificate filename="Certificates\privateKeystore.pfx" password="changeit" />
```

Modify app.config

The NuGet-package adds a config-section automatically:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>
    <section name="peppol.certificates"
type="STARTLibrary.src.eu.peppol.start.security.configuration.CertificatesConfigurationSection,
STARTLibrary45" />
  </configSections>
```

If you are using test-certificates in stead of live you have to change thumbprints used for validation:

```
      <add name="MyRootCertificates"
           rootCACertificateThumbprint="C55F371D9B3C3A54A06FD51E02E43F9E90F6D65C"
           intermediateSmpCACertificateThumbprint="9C200AB2044F67BA89D9ADE2180CCDE878639470"
           intermediateAcessPointCACertificateThumbprint="696D7543E15F84A32A1121531D0551BCB3AAEA50" />
    </validation>
```

Use the same endpointname in the peppol.certificates-section, WCF-configuration(system.servicemodel->client->endpoint->name) and when creating your EHFUtility-instance.

```
    <clientCredentials>
      <add endpointName="SecurePeppolClient">
```

If you are testing against a specific endpoint/access point, fill in that servers public key in the serviceCertificate-element:

```
             <serviceCertificate
encoded="MIIESzCCAzOgAwIBAgIQVrdpzaiwUuXBpr+naizO3DANBgkqhkiG9w0BAQsFADBX&#xD;&#xA;MQswCQYDVQQGEwJESzEnMCU
GA1UEChMeTkFUSU9OQUwgSVQgQU5EIFRFTEVDT00g&#xD;&#xA;QUdFTkNZMR8wHQYDVQQDExZQRVBQT0wgQUNDRVNTIFBPSU5UIENBMB4
XDTE0MDEw&#xD;&#xA;ODAwMDAwMFoXDTE2MDEwODIzNTk1OVowUDELMAkGA1UEBhMCTk8xFzAVBgNVBAMM&#xD;&#xA;DkFQUF8xMDAwM
DAwMDY0MSgwJgYDVQQKDB9GeWxrZXNtYW5uZW4gaSBTb2duIG9n&#xD;&#xA;IEZqb3JkYW5lMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AM
IIBCgKCAQEAqNLdts3A&#xD;&#xA;xyYW98n-REMOVED-
&#xD;&#xA;BgNVHQ4EFgQULr75Ch041QO4LsB0aNwUczyYrXowNwYIKwYBBQUHAQEEKzApMCcG&#xD;&#xA;CCsGAQUFBzABhhtodHRwOi
8vcGtpLW9jc3Auc3ltYXV0aC5jb20wEwYDVR0lBAww&#xD;&#xA;CgYIKwYBBQUHAwIwDQYJKoZIhvcNAQELBQADggEBAIAI5Y2TD1Ld61
XzCHpRWbQL&#xD;&#xA;1rBiP0okp3KBugtdfUJJ76UArUDwajsSMRetPYmRNgZYY1ix2FfPcJ4wBJf4i85d&#xD;&#xA;jXSJUsjQmqbZ
0LBoUqYrxGJdbK0Gjn38zwm4z3ucjKHMjTIn3cRE9TS74q2RakNw&#xD;&#xA;nA1gjiNKKXtBN2VIHXUHRzDKZ6Hd42XhJgj477rwn92U
0LOlZU3sLgmSLqs5GCbF&#xD;&#xA;JXVri40NcliDCre3bQe2cUy26fdd6FnC2Z1S2sZ9wFQ7tuWjG6cdfJ301Pdjh7RV&#xD;&#xA;wO
zV1+VsORHVh8rgJKWkgVsDErunJVwEPeDeIL0KxUR0BvLU3ygwQR3HaW6iXmQ="/>
      </add>
    </clientCredentials>
  </peppol.certificates>
```