

Web Services

Product version:	4.61
Document version:	1.0
Document creation date:	07-06-2006
Document last saved:	02-08-2006

Purpose

The contents of this document are directly taken from the EPiServer SDK. Please see the SDK for further technical information about EPiServer.

Table of Contents

INTRODUCTION	2
WEB SERVICES REQUIREMENTS	2
CONFIGURE EPISERVER TO ENABLE BASIC AUTHENTICATION	3
STEP 1: CONFIGURE INTERNET INFORMATION SERVER	3
STEP 2: CONFIGURE ACCESS RIGHTS IN ASP.NET	4
STEP 3: ACTIVATE BASICAUTHENTICATION MODULE.....	4
STEP 4: TEST THE SETUP	4
TROUBLESHOOTING	4
I CANNOT GET PAST THE WINDOWS LOGIN PROMPT	4
I GET A FORMS LOGIN PROMPT, AFTER THE WINDOWS LOGIN PROMPT.....	4
CREATING YOUR OWN WEB SERVICES	5
CREATING A UTILITY SERVICE	5
CONSUMING THE WEB SERVICE	6
COMMUNICATION WITH .NET SOAP CLIENT	7

Introduction

The EPiServer sample site includes the following Web services ready for use.

Authenticate.asmx

This is used primarily by EPiServer for authentication requests when connecting one or more EPiServer sites.

CacheService.asmx

This Web service is used primarily to synchronize the cache between different EPiServer sites (in a Web farm or loosely connected sites.)

DataFactoryService.asmx

This is a powerful Web service that contains methods to retrieve, add, delete and search for information from the EPiServer site.

PageMirroringService.asmx

This is used primarily when mirroring content in EPiServer.

Note The implementation of these services is compiled into episerver.dll. Do not include the .asmx files in your Visual Studio.NET project as you might accidentally create new code-behind files for the services. Click **No** if Visual Studio.NET asks to create code-behind files for the existing .asmx files. By creating your own Web services, you can extend the available functionality of EPiServer and your site. If you place your own services inside the WebServices directory, they will be secured like the built-in EPiServer Web services.

Web Services Requirements

EPiServer is installed, by default, with forms authentication. Web service clients cannot communicate with a Web service that uses forms authentication, as the authentication occurs through an HTML user interface meant for visitors on the Web site. You must use Integrated Windows authentication or follow the steps in Configure EPiServer to Enable Basic Authentication to emulate Basic authentication if you want to use both forms authentication and Web services on the same site.

The standard installation of EPiServer contains a directory called /WebServices. This directory is protected by default in the web.config file like this:

```
<location path="/WebServices">
  <system.web>
    <authorization>
      <allow roles="WebServices,Administrators" />
      <deny users="*" />
    </authorization>
  </system.web>
</location>
```

It is recommended to use a dedicated user for authenticating Web service clients, like the "WebServices" user shown above. When using forms authentication on the Web site, the section shown above will instruct ASP.NET to redirect all requests for .NET-handled files (.aspx, .asmx etc.) to the login form. When you are writing a client program to communicate with any of the existing Web services (or any new ones you write) in the /WebServices directory, you do not want the program to be redirected to a page other than the Web service .asmx page. Your client program will not know how to authenticate using the returned HTML, and it will typically throw an exception.

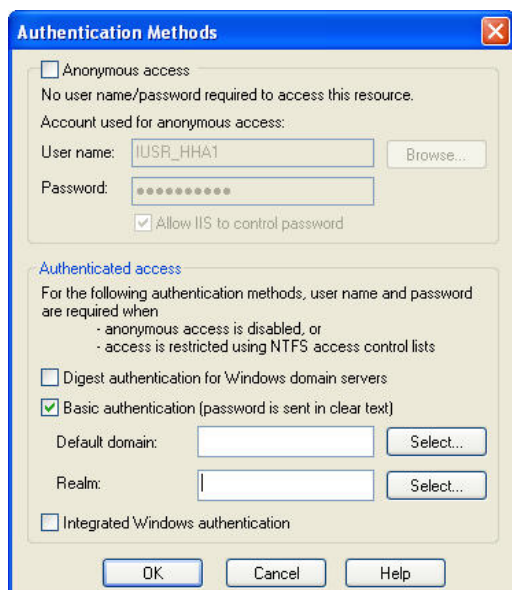
There is a workaround for this in EPiServer if you want to keep forms authentication and still want to expose the Web services.

Configure EPiServer to Enable Basic Authentication

Web Services cannot authenticate against a forms-authenticated site, because the forms authentication login window requires user interaction. This chapter describes how to configure and set up EPiServer to enable basic authentication, normally only supported when using Windows authentication, on parts of the Web site.

Step 1: Configure Internet Information Server

1. Open Internet Information Services Manager on the Web server and select the /WebServices folder on your EPiServer Web site.
2. Right-click and select **Properties**. Under the **Directory Security** tab, click **Edit**.
3. The authentication options must be configured for Basic Authentication only. Otherwise automatic authentication will not occur.



Step 2: Configure Access Rights in ASP.NET

Edit the Web configuration file, web.config, in the EPiServer root directory. Make sure that the Web service account is allowed access in the WebServices folder.

```
<location path="WebServices">
  <system.web>
    <authorization>
      <allow users="DOMAIN\MyWebServiceUser" />
      <deny users="*" />
    </authorization>
  </system.web>
</location>
```

Replace DOMAIN with the name of the domain or local machine where the user account was created.

Step 3: Activate BasicAuthentication Module

The BasicAuthentication http module will translate basic authentication requests on-the-fly to forms-authenticated cookies. Make sure that web.config has the BasicAuthentication filter defined under the httpModules section.

```
<httpModules>
<add name="BasicAuthentication" type="EPiServer.Security.BasicAuthentication,
EPiServer" />
```

Step 4: Test the Setup

1. Open a Web browser and enter the URL to a Web service on your Web site, for example: <http://localhost/WebServices/DataFactoryService.asmx>. You will receive a standard Windows login pop-up window.
2. Enter the WebServiceUser account information. If everything is working, you should see the Web Service definition page

Troubleshooting

I Cannot Get Past the Windows Login Prompt

Make sure that the WebService user has access to log on to the server. Normally you have to either make this user a member of the Users group, or enable the local policy "Log on locally". Also check that the user has NTFS access to the WebServices folder. Most often you have to restart Internet Information Services after changing security settings in Windows such as policy or NTFS rights.

I Get a Forms Login Prompt, After the Windows Login Prompt

Enter the username and password in the forms authentication prompt. If you get through, verify that the BasicAuthentication filter is installed correctly.

1. If you cannot get past the forms login prompt, verify that forms authentication with Windows accounts actually works. Log on to Edit mode with a local administrator account.
2. If this doesn't work, make sure the ASPNET account on Windows 2000 has the "Act as part of the operating system" policy. (This should not be a problem on Windows Server 2003).

If everything apart from the forms login to the WebServices folder works, verify the web.config settings. For example, if you are logging on with COMPUTER\MyWebServiceUser, verify that the identical name was set in

web.config. Skipping domain or adding the wrong domain could be the problem. Most often you have to restart Internet Information Services after changing security settings in Windows, such as policy or NTFS rights.

Creating Your Own Web Services

If you create your own Web services, place them in the /WebServices folder to have the same security settings as the built-in Web services. This is especially important, if you need to use forms authentication on your site. All the information you can access through the EPiServer API can also be exposed through Web services.

Note If you access an EPiServer Web service using .NET, which has been protected by the **BasicAuthentication** HttpModule, you need to set `PreAuthenticate = true` for the .NET Web service proxy code to authenticate each request.

Creating a Utility Service

The following Web service makes all EPiServer configuration settings available for external clients.

Note This also exposes the connectionstring, usernames and passwords.

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Web;
using System.Web.Services;
using System.Xml;
using System.Text;
using System.IO;
using EPiServer;
using EPiServer.Core;

namespace development.
{
    /// <summary>
    /// Utility members for EPiServer
    /// </summary>

    [WebService(Namespace="http://episerver.com/episerversample/webservices/",
        Description="Utility functions for EPiServer, giving you
        information about the site.")]
    public class EPiServerUtil : System.Web.Services.
    {
        [WebMethod(Description="Returns the servers time according to
        DateTime.Now()")]
        public DateTime ServerTime()
        {
            return DateTime.Now;
        }

        [WebMethod(Description="Returns all configuration settings for this
        site as XML.")]
        public string ConfigurationXml()
        {
            System.Collections.Specialized.NameValueCollection oSettings;

            StringBuilder oBuilder = new StringBuilder();
            StringWriter oTextWriter = new StringWriter(oBuilder);
            XmlTextWriter writer = new XmlTextWriter(oTextWriter);
```

```

        // Build the XML
        writer.WriteStartDocument();
        writer.WriteStartElement("episerverconfig");
        writer.WriteAttributeString("version", Global.EPConfig.Version );
        writer.WriteStartElement("values");

        oSettings = Global.EPConfig.ConfigFile.AllAppSettings;

        for (int i = 0; i < oSettings.Count; i++)
        {
            writer.WriteStartElement("value");
            string[] keyvalue = oSettings.GetValues(i);
            writer.WriteElementString("key", oSettings.Keys[i]);
            writer.WriteElementString("value", string.Join(",",
keyvalue));
            writer.WriteEndElement();
        }
        writer.WriteEndElement();
        writer.WriteEndElement();
        writer.WriteEndDocument();

        writer.Flush();
        writer.Close();
        oTextWriter.Close();

        return oBuilder.ToString();
    }
    public EPiServerUtil()
    {
        //CODEGEN: This call is required by the ASP.NET Web Services
Designer
        InitializeComponent();
    }

    //Required by the Web Services Designer
    private IContainer components = null;

    private void InitializeComponent()
    {
    }

    protected override void Dispose( bool disposing )
    {
        if(disposing && components != null)
        {
            components.Dispose();
        }
        base.Dispose(disposing);
    }
}
}
}

```

Consuming the Web Service

1. Create a new C# Console project in Visual Studio.NET
2. Add a Web Reference to the newly created Web Service
3. Use the code below for the implementation. (Remember to change the username, password and domain.)

```

using System;
using System.Text;
using System.Xml;
using System.Net;

```

```

// Change this to the namespace of your webservice
using ConsoleTest.localhost;

namespace ConsoleTest
{
    class ConsoleTest
    {
        [STAThread]
        static void Main(string[] args)
        {
            StringBuilder allSettings = new StringBuilder();

            EPiServerUtil wsUtil = new EPiServerUtil();
            wsUtil.Credentials = new NetworkCredential("john", "doe", "DOMAIN");

            wsUtil.PreAuthenticate = true;

            string settingsXml = wsUtil.ConfigurationXml();
            XmlDocument xmlDoc = new XmlDocument();
            xmlDoc.LoadXml(settingsXml);
            XmlNodeList nodes = xmlDoc.SelectNodes("/episerverconfig/values/value");

            foreach (XmlNode node in nodes)
            {
                allSettings.AppendFormat("{0} = {1}\r\n",
                    node.SelectSingleNode("key").InnerText,
                    node.SelectSingleNode("value").InnerText);
            }

            Console.Out.Write(allSettings.ToString());
        }
    }
}

```

Communication with .NET Soap Client

When communicating with EPiServer using a .NET soap client, set the property `SoapHttpClientProtocol.PreAuthenticate` to **true** to make sure that the username and password are sent to the server at every request, instead of using the default behavior that relies on connection keep-alive and access-denied round-trips.

The main reason is that if the client and server are using connection keep-alive without storing cookies, the BasicAuthentication filter may not be able to capture following requests that reuse a previously authenticated connection.

Copyright © EPiServer AB. EPiServer is a registered trademark of EPiServer AB. Other product and company names mentioned in this document may be the trademarks of their respective owners.

The document may be freely distributed in its entirety, either digitally or in printed format, to all EPiServer users. Changes to the content or partial copying of the content may not be carried out without permission from EPiServer AB:

EPiServer AB
Finlandsgatan 38
SE-164 74 Kista
Sweden

Changes are periodically made to the document and these will be published in new editions of the document. EPiServer AB reserves the right to improve or change the products or programs included in this document at any time.