ELMS-Connector Installation

This document should outline the basic functionality of ELMS-Connector and the required steps for setup. ELMS-Connector is an open source project developed by Daniel Hoelbling, it is licensed under the [Apache License Version 2.0](http://www.apache.org/licenses/LICENSE-2.0.html) and is available through GitHub at: <http://github.com/Tigraine/elms-connector>

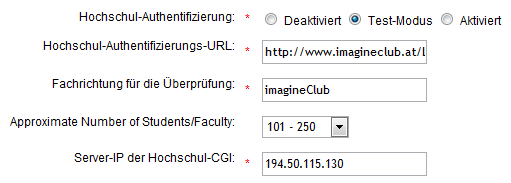
This document assumes you are familiar with the ELMS Software distribution platform by Microsoft and have admin rights to ELMS.

This document targets [ELMS-Connector version 1.0.0.0](http://github.com/Tigraine/elms-connector/tree/v1.0.0.0).

# ELMS Campus Authentication

To explain what ELMS-Connector does, a little knowledge about the ELMS Campus Authentication is required.

ELMS Campus Authentication depends on the user coming to the Campus system with a Token that has been assigned by ELMS. This is done by the user opening the Microsoft ELMS Campus page with his browser, and being redirected (with the added token) to your Campus Authentication page. You can specify in your ELMS Administration what URL the user should be redirected to under: User Account Management -> Integrated Campus Authentication



Once the User got his token from ELMS you have to authenticate the user locally against your database. If you find the user has sufficient privileges to access the ELMS-Store you then have to create a server-side HTTP request to ELMS, to inform ELMS of the newly opened session. Once these steps have been done, you can redirect the client to the ReturnUrl initially passed to you alongside the Token.

# Purpose of ELMS-Connector

The idea behind ELMS-Connector is to abstract away all of the handshaking, requiring the implementor only to take care of the authentication against the local database.

# Obtaining ELMS-Connector

ELMS-Connector is available as source code or as precompiled binaries. Both target the .NET 3.5 framework.

The source code is available through [GIT](http://git-scm.com/) by doing a

git clone git://github.com/Tigraine/elms-connector.git

Or as zip download:

<http://github.com/Tigraine/elms-connector/archives/master>

If you prefer to not compile ELMS-Connector yourself you can also use the precompiled packages here:

<http://github.com/Tigraine/elms-connector/downloads>

# Building ELMS-Connector

ELMS-Connector is written entirely in C# targeting the .NET Framework v3.5, to compile (and deploy) ELMS-Connector you need to install [.NET Framework v3.5 SP1](http://www.microsoft.com/downloads/details.aspx?FamilyId=AB99342F-5D1A-413D-8319-81DA479AB0D7&displaylang=en) on your machine and have [Microsoft Windows Powershell](http://www.microsoft.com/powershell) installed.

Once .NET and Powershell are both installed you should be able to simply execute the ClickToBuild.bat located in your ELMS-Connector source directory.

Due to security restrictions in default Powershell installations it may be necessary to execute the following Powershell command for the build script to run:

Set-ExecutionPolicy remotesigned

After a successful build you will find a folder called /release inside your ELMS-Connector source directory. It will contain one file named ElmsConnector-v1.0.0.0.zip, this file contains all necessary files to install ELMS-Connector.

# Installation

Installation is pretty straight forward, but still non-trivial. Here the necessary steps:

## Reference assembly

1. Reference ElmsConnector.dll from your existing (or blank) ASP.NET application.
2. Place Login.htm and elms.xml in your application’s root directory.

## Add HttpHandlers

Next open up your web.config and add the ELMS-Connector HttpHandler :

<httpHandlers>  
 <remove verb="\*" path="\*.asmx"/>  
 <add verb="\*" path="\*.axd" type="ElmsConnector.ElmsHandler, ElmsConnector"/>  
</httpHandlers>

This means that all requests to \*.axd on your server will be handled by ELMS. If you have other libraries installed that require the same extension you can either limit the path to Login.axd and VerifyUser.axd or use another extension altogether (in that case make sure to configure ELMS-Connector properly for the usage of another file extension)

## Write AuthenticationService

In order for ELMS-Connector to know if a user is allowed to log in or not you have to provide an adequate implementation of the interface IAuthenticationService or if you plan on allowing session login IExtendedAuthenticationService.

IAuthenticationService defines only one method:

public interface IAuthenticationService

{

bool AuthenticateUser(string username, string password);

}

You now have to write an implementation of IAuthenticationService that returns true if username/password do match and false if the user should be denied access.

## Modify elms.xml

After having a working IAuthenticationService implementation you need to tell ELMS-Connector where to find that class. You do that through the elms.xml file where you have to set the type attribute on the component with the id AuthenticationService. An example:

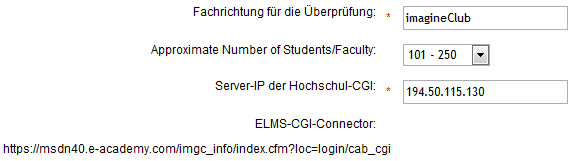
<component

id="AuthenticationService"

service="ElmsConnector.IAuthenticationService, ElmsConnector"

type="YourNamespace.YourClass, YourAssemblyName " />

You also have to set the cgiConnector URL for your ELMS installation. You can copy&paste the URL from your ELMS Administration site:



You copy the ELMS-CGI-Connector URL straight into your elms.xml file like to the <cgiConnector> node:

<component

id="ElmsSessionRequestService">

<parameters>

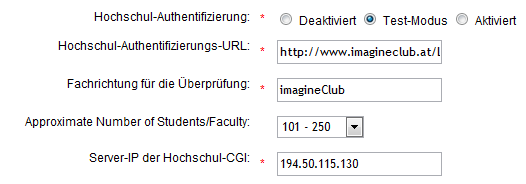
<cgiConnector>https://msdn60.e-academy.com/imgc\_info/index...</cgiConnector>

</parameters>

</component>

## Configure ELMS

You now need to tell Microsoft’s ELMS System where to find the integrated campus authentication page provided by ELMS-Connector. To do so you simply visit your administration and set these marked values to you web-application’s URL:



The Campus-Authentication-URL would look like this:

http://www.yourserver.com/Login.axd

# Testing ELMS-Connector

This should be sufficient to get ELMS-Connector set up if you are using the default file extension (axd). You can now try to access your ELMS Portal and should be redirected to a blank page prompting you for username and password. Upon login you should find yourself logged into the ELMS Shop.

# Customizing ELMS-Connector

To change the way your login page looks simply edit the file Login.htm that you placed in your web application’s root directory. Make sure to keep the placeholders intact ($ERROR$...)

## Setting a different file extension

If .axd is already taken or you don’t want that you need to adjust your HttpHandler line in your web.config and you need to add the following to your elms.xml

<component id="FileExtensionProvider">

<parameters>

<extension>yourextension</extension>

</parameters>

</component>

Your custom extension has to go into the <extension> node and ELMS-Connector will pick that up.  
Note: Changes to elms.xml are only loaded when the application restarts up. (You can restart your ASP.NET web application by touching the web.config).