Interzoic Media
Business and Community Web Development

1775 W. State St. Suite 371
Boise, Idaho 83702 USA
Phone: 208.713.5974
www.interzoic.com

Interzoic Single Sign-On for DNN Portals

Version 2.3.0

Table of Contents

Download Packages

Interzoic.SSO.Package.02.03.00_UnZipMe.zip - Master Zip file that contains all of the following files:

Interzoic_Single_Sign-On_User's_Guide.pdf – User's Guide (this document)
Interzoic.SSO.02.03.00_Install.zip - SSO DNN Module Install
Interzoic.SSO.ClientApplication.02.03.00.zip - Testing Harness
Interzoic.SSO.NET.Client.API.02.03.00.zip - API Class Library for .Net development
Interzoic.SSO.NET.Client.API.02.03.00_Documentation.chm – Help file for the API Library

Introduction

The Interzoic Single Sign-On application installs in a DNN Software portal framework and provides User account management and auto-login/single sign-on (SSO) services from a remote website to the host DNN portal. The remote server utilizes REST based Web Services provided by the Interzoic SSO to query and set User account, profile and role data in the DNN portal. The Web Services also provide a secure Login Token that the remote website can include in a URL string that will allow an authorized User to auto-authenticate and auto-login to the DNN portal. This guide exercises the setup and configuration of the SSO module that has been installed on the http://demo.accordlms.com/ home page.

The Single Sign-On Tips blog article provides useful tips on how configure your site for better SSO integration: hide Login controls; redirect for secure content and redirect after Logout.

DNN Server Requirements

- DNN Version 7.1.0 or higher (or DNN 6.2.4+ for SSO 1.6)
- Full and Medium Trust Security Levels are supported
- .NET 4.0 or higher

Release Notes

Version 1.1.0

- If you are upgrading from a previous version, you'll have to manually delete from /bin these two files: Interzoic-SSO.dll and Interzoic-SSO.Shared.dll
- New UserRoleGet service
- New .NET client API
- Updated documentation and sample code

Version 1.2.0

- Small updates

Version 1.3.0

- Added "UserIsOnLine" service.
- Updated documentation, .NET API and ClientApplication.

Version 1.4.0

- Added "Codebase" licensing option.

Version 1.5.0
- Added an HTML Form “POST” option for auto-login (and creation) of portal users.

Version 1.6.0
- Requires .NET 4.0+ and DNN 6.2.4+.
- Supports Windows Azure.
- Roles include also the RoleType, SecurityMode and Status properties.

Version 2.0.0
- New security schema. Instead of DNN users, the SSO now utilizes its own users set to authorize web services requests.
- Updated for DNN 7.1.0 or later.
- More digits for the “SSO Token Timeout” setting.
- Improved identification of the current (local) Url for the license key.

Version 2.1.0
- New service to ‘hard delete’ users
- The web services error messages are now more informative
- Support for usernames including a '\' character (replace with ‘^’)
- Requires .NET 4.0+ and DNN 7.0.6+

Version 2.2.0
- New “Authorize base URL” to avoid domain mismatch issues.

Version 2.3.0
- New functionality to avoid errors when the URI contains a blank space
- Updated documentation

SSO Auto-Login & Administrative Module

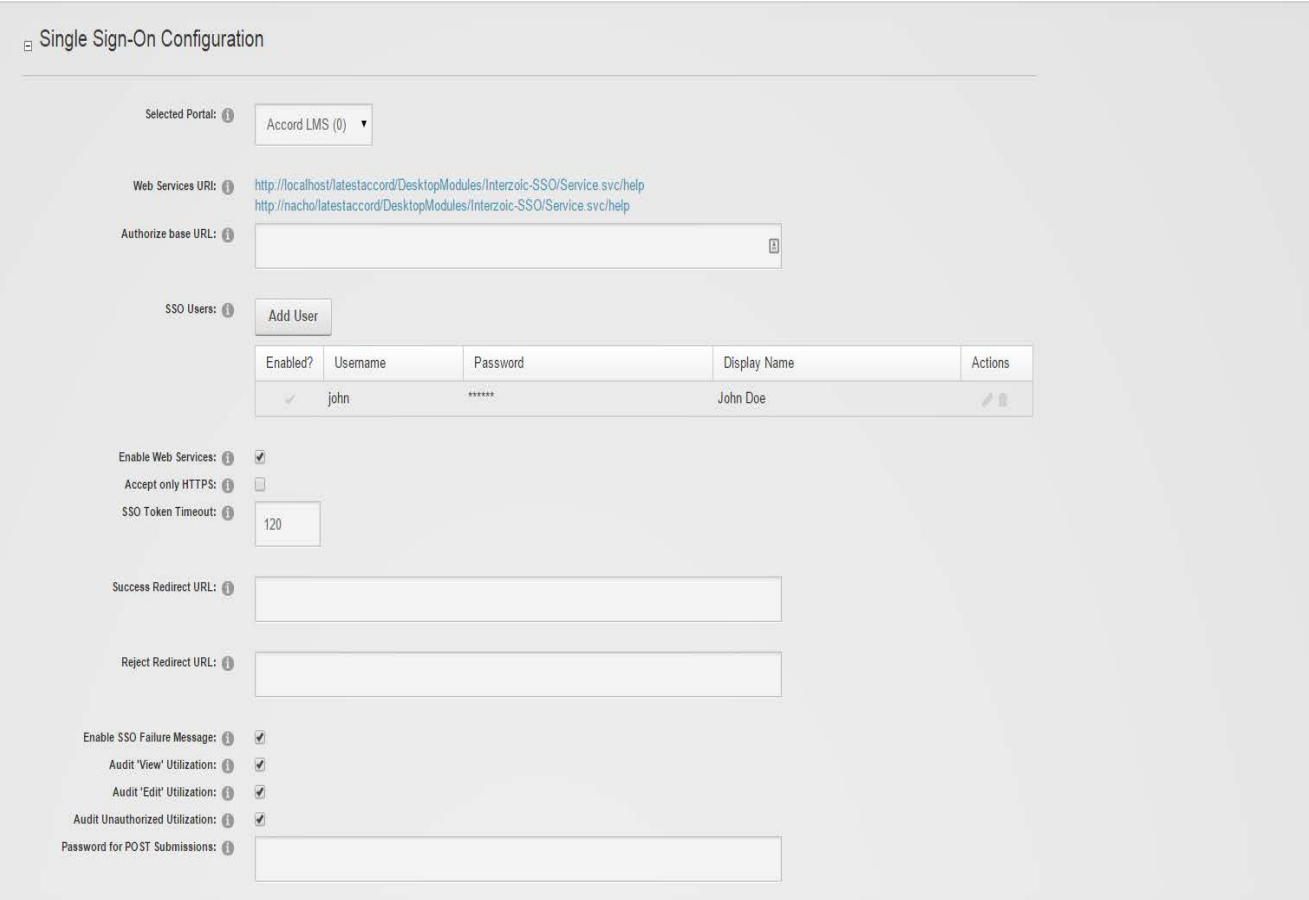
The SSO application is installed in a DNN portal framework using the standard procedure for installing a DNN module. Once installed, the SSO Auto-Login module can be added to any page that you want to be a target landing page from the remote server. The administrative/configuration options will be secured, only available for Users with “Edit” permissions to the module. The module does not have a visible UI, so non-Admin Users will not be able to detect its presence.

Quick Start

Installation and Configuration

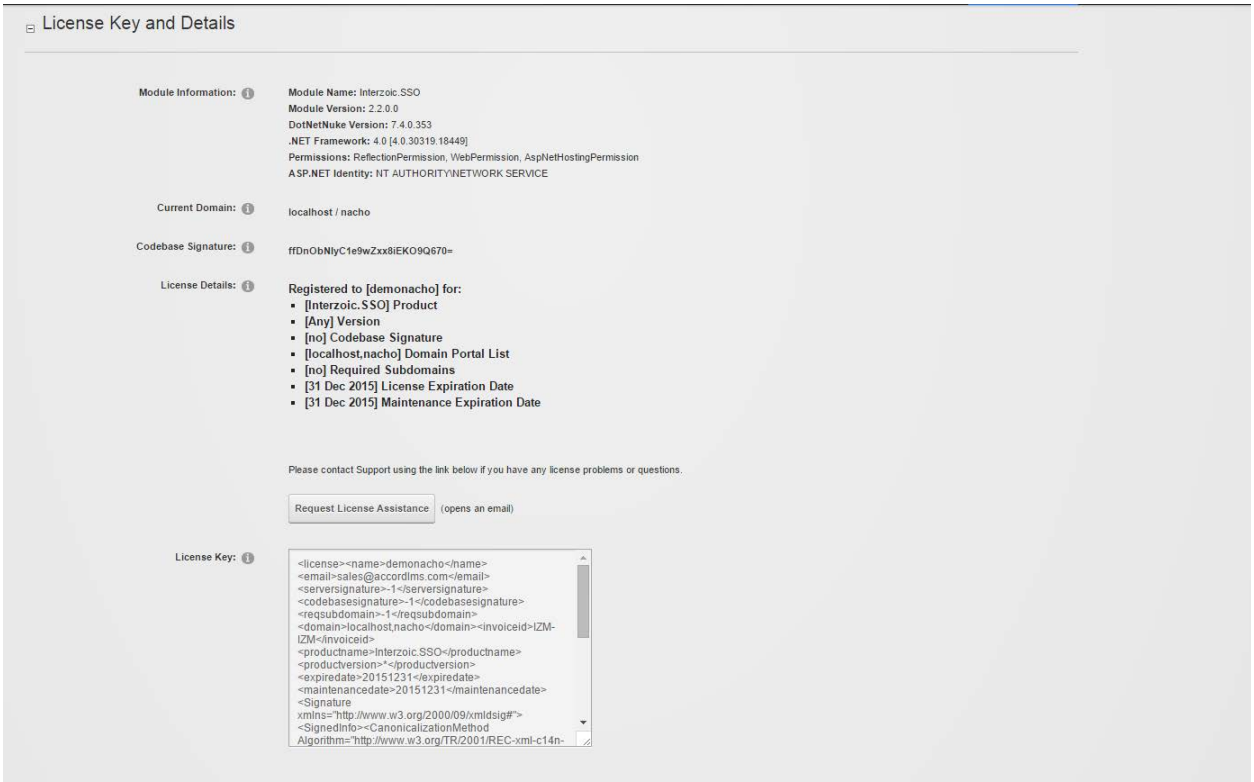
The Interzoic Single Sign-On application installs in the DNN Software portal framework using the standard module installation process. Once installed, add the module to any page that you want to use as a “Landing” page. Once SSO Administration options are set, the web services will be functional.

Access the “SSO Configuration” UI from the Action Menu drop down.



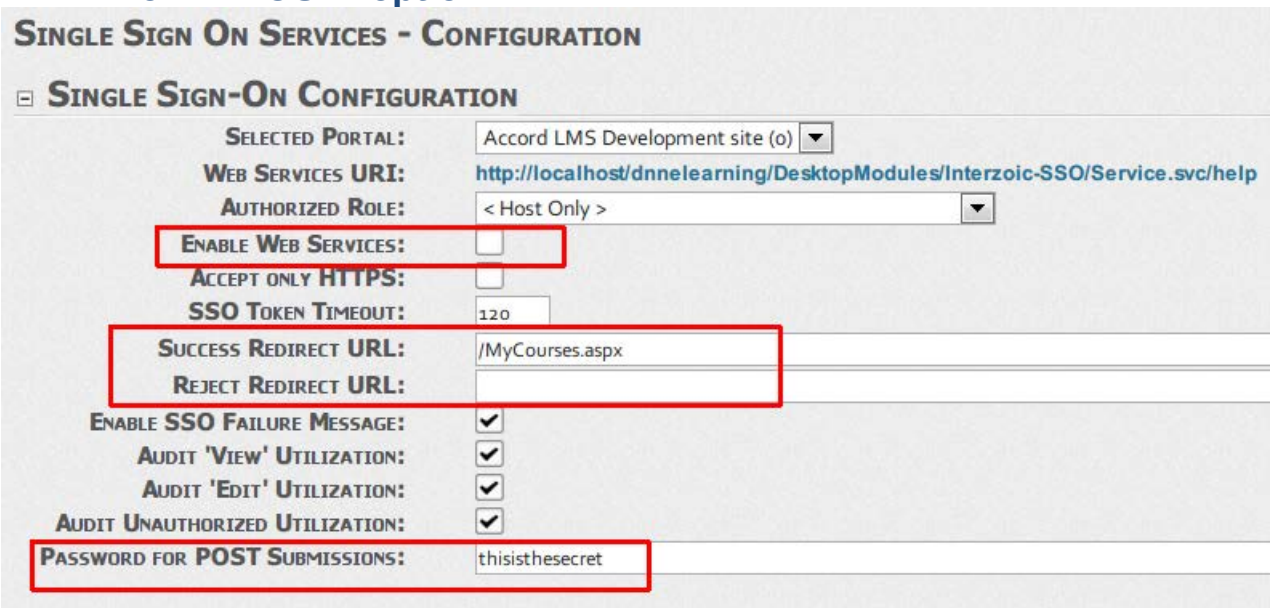
- **Selected Portal:** Portal Selection DropDown (Host can access all portals)
- **Web Services URI:** Available URIs based on Portal Aliases
- **Authorize base URL:** Alias URL for correct hash authorization when calling the service with a URI that differs with the domain name
- **SSO Users:** For the currently selected portals, the list of available SSO user accounts, which (when enabled) will be allowed to perform web services requests. You can add/edit/delete SSO user accounts from this page.
  NOTE: By default no account will be created for you. You must create at least one SSO user in order to perform web services requests.
- **Enable Web Services:** Enable/Disable
- **Accept only HTTPS:** Enable/Disable
- **SSO Token Timeout:** Seconds that Token will remain valid
- **Success Redirect URL:** URL to redirect to if successful login
- **Reject Redirect URL:** URL to redirect to if rejected login
- **Enable SSO Failure Message:** Enable/Disable
- **Audit 'View' Utilization:** Enable/Disable
- **Audit 'Edit' Utilization:** Enable/Disable
- **Audit Unauthorized Utilization:** Enable/Disable
- **Password for POST Submissions**: If a value has been set, the auto-login via POST will be enabled.

License Configuration



On the same “SSO Configuration” UI, under the "License Key and Details" section, you can see the current 'Server Signature' which you'll need to provide when requesting a 'Codebase' license. For regular licenses keys, you should provide us the 'Current Domain' value.
To apply/install your license, simply paste its content in the 'License Key' text box.

HTML Form “POST” option



You can leave the “Enable Web Services” unchecked, if you’ll not be using them (that is, you’ll just use the POST feature).
There is also a new configuration setting (the password for the HASH validation).

If you leave the “Password for POST Submission” blank, then this feature will be disabled. This is the default configuration.

Sample HTML Form

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>SSO POST Test</title>
  </head>
  <body onload="gotoSITE.submit()" style="text-align:center">
       <h1>SSO POST Test</h1>
       <form name="gotoSITE" action="http://localhost/dnnelearning/en-us/sso.aspx"
method="POST">

       <!-- thisisthesecret -->
         <input type="hidden" name="HASH"  value="783583dce8bcb6cb4f0c0379fcdc01d7"/>

         <!-- "yyyy-MM-ddTHH:mm:ss"  '0001-01-01T00:00:00 / 2010-04-30T00:00:00 (without
milliseconds) -->
         <input type="hidden" name="TIMESTAMP"  value="2013-02-15T22:51:58"/>

         <input type="hidden" name="SUCCESSURL"  value="http://localhost/dnnelearning/en-
us/mycourses.aspx"/>
         <!-- input type="hidden" name="REJECTURL"
value="http://localhost/dnnelearning/default.aspx"/ -->

         <input type="hidden" name="USERNAME"  value="johndoe2"/>
         <input type="hidden" name="PASSWORD"  value="john'sPasswd"/>
         <input type="hidden" name="FIRSTNAME"  value="John II"/>
         <input type="hidden" name="LASTNAME"  value="Doe"/>
         <input type="hidden" name="EMAIL"  value="johndoeII@example.com"/>

       </form>
  </body>
</html>
```

About the redirect URLs

SUCCESSURL and REJECTURL are optional. If they are left blank or they are not present, the module will use the settings configured on its ‘Configuration’ page.
If the rejectURL is empty also on this configuration page, you get message on the current page (this is good for ‘debugging’).

About the HASH

This is how the string to be hashed is built and validated:

```
Dim hash As String = Request.Form("HASH")
Dim timeStamp As String = Request.Form("TIMESTAMP")
```

```
        Dim successURL As String = Request.Form("SUCCESSURL")
        Dim rejectURL As String = Request.Form("REJECTURL")

        Dim userName As String = Request.Form("USERNAME")
        Dim userPassword As String = Request.Form("PASSWORD")
        Dim userFirstName As String = Request.Form("FIRSTNAME")
        Dim userLastName As String = Request.Form("LASTNAME")
        Dim userEmail As String = Request.Form("EMAIL")

        Dim stringToEncrypt = String.Format("{0}{1}{2}{3}{4}{5}{6}{7}", _
                                            postPassword, _
                                            timeStamp, _
                                            successURL, _
                                            rejectURL, _
                                            userName, _
                                            userPassword, _
                                            userFirstName, _
                                            userLastName)

        Dim encryptedString As String = MD5Crypt.MD5(stringToEncrypt)
        Dim isHasValid As Boolean = (hash = encryptedString)
```

NOTE: The algorithm is the same as PHP's MD5 function). A .NET implementation of this HASH function is as follows (ref. http://www.spiration.co.uk/post/1203/MD5-in-C%23---works-like-php-md5()-example):

```
public static string MD5(string password) {
   byte[] textBytes = System.Text.Encoding.Default.GetBytes(password);
   try {
      System.Security.Cryptography.MD5CryptoServiceProvider cryptHandler;
      cryptHandler = new
System.Security.Cryptography.MD5CryptoServiceProvider();
      byte[] hash = cryptHandler.ComputeHash (textBytes);
      string ret = "";
      foreach (byte a in hash) {
         if (a<16)
            ret += "0" + a.ToString ("x");
         else
            ret += a.ToString ("x");
      }
      return ret ;
   }
   catch {
      throw;
   }
}
```

About the TIMESTAMP

It must be the current UTC date time in the "yyyy-MM-ddTHH:mm:ss" format (without milliseconds).
To make the auto-login more robust, the module (SSO) will validate this TIMESTAMP is within 15 minutes of the Web Server UTC system time when the request is received. If it is not, the action (auto-login and/or create the user) will fail and an error will be thrown.

About the User's data

If you know the user already exists, there is no need to include the PASSWORD, FIRSTNAME, LASTNAME and EMAIL. These are utilized to create the new user when the referenced "USERNAME" does not exist.

SSO Web Services

As part of the Interzoic SSO module installation, REST web services are deployed but not enabled. Once SSO Administration options are set, the web services will be functional.

You can get a live overview and information about the provided Services by browsing to their 'help' page: http://<root.url>/DesktopModules/Interzoic-SSO/Service.svc/help

The URI provides the most current information about the available services and how to use them.

Service help page

IService: UserGet
Thu, 29 Jul 2010 3:35 PM

UriTemplate	http://demo.accordlms.com/DesktopModules/Interzoic-SSO/Service.svc/portal/{portal}/users/{username}
Method	GET
Response Format	Xml
Response Schema	http://demo.accordlms.com/DesktopModules/Interzoic-SSO/Service.svc/help/UserGet/response/schema
Response Example	http://demo.accordlms.com/DesktopModules/Interzoic-SSO/Service.svc/help/UserGet/response/example
Description	For a given PortalID and Username, returns the User's Account.

IService: UserDelete
Thu, 29 Jul 2010 3:35 PM

UriTemplate	http://demo.accordlms.com/DesktopModules/Interzoic-SSO/Service.svc/portal/{portal}/users/{username}
Method	DELETE
Request Format	xml or json
Request Schema	http://demo.accordlms.com/DesktopModules/Interzoic-SSO/Service.svc/help/UserDelete/request/schema
Request Example	http://demo.accordlms.com/DesktopModules/Interzoic-SSO/Service.svc/help/UserDelete/request/example
Response Format	Xml
Response Schema	http://demo.accordlms.com/DesktopModules/Interzoic-SSO/Service.svc/help/UserDelete/response/schema
Response Example	http://demo.accordlms.com/DesktopModules/Interzoic-SSO/Service.svc/help/UserDelete/response/example
Description	For a given PortalID, UserID and UserInfoDataContract, creates or updates the User's Account.

Authorization and Authentication

The authorization and authentication approach for the SSO module was drawn from Amazon S3 REST web services.

Each web service call must include authentication information. If the request is valid (authenticated and authorized) the service action is processed, otherwise **403 (Forbidden)** HTTP error code will be returned, along with a **text/xml** response message with the following structure:

<Error><Code>**ErrorCode**</Code><Message>**Error Message**</Message><ServerTime>**UTCDateTime**</ServerTime></Error>

Example:
<Error><Code>**InvalidProtocol**</Code><Message>**You must use HTTPS protocol.**</Message><ServerTime>**2010-06-21T22:04:53**</ServerTime></Error>

The possible error codes are:

- **RequestTimeTooSkewed**: the client time-stamp included with an authenticated request is not within 15 minutes of the Web Server UTC system time.
- **InvalidPortalID**: The PortalID in the Authorization header is not valid.
- **DisabledServices**: For the given portal, the web services are not enabled from the administrative/configuration UI.
- **InvalidProtocol**: The request was HTTP but the portal was configured to accept only HTTPS Web Services requests.
- **InvalidRequestHeaders**: Some data is wrong regarding the Authorization or x-izm-date headers.
- **InvalidUsername**: The Username in the Authorization header is not valid.
- **InvalidSignature**: The Signature (hash) in the Authorization header does not match the hash computed at the server side.
- **Unauthorized**: The User configured in the Authorization header does not have rights to access the service.
- **InvalidLicense**: The License configured for this Portal (from the administration UI) is not valid.
- **GenericError**: Any unclassified error.

Authenticating a Request

When accessing the REST Web Services, you must provide the following items so the request can be authenticated:

- **DNN Username and PortalID**: Your DNN account is identified by your DNN Username in a given PortalID. Note: Host/SuperUser accounts can provide any valid PortalID.
- **Signature**: Each request must contain a valid request Signature, or the request is rejected. A request Signature is calculated using your DNN PortalId +Username + Password.
- **Time stamp**: Each request must contain the date and time the request was created, represented as a string in UTC.
    - The format of the value of this parameter is: **yyyy-MM-ddTHH:mm:ss**. i.e.: 2010-04-30T21:15:55
    - A valid time stamp (using a **x-izm-date header)** is mandatory for authenticated requests.
    - The client time-stamp must be within 15 minutes of the Web Server UTC system time when the request is received. If not, the request will fail with the RequestTimeTooSkewed error status code.

  The intention of these restrictions is to limit the possibility that intercepted requests could be replayed by an adversary. For stronger protection against eavesdropping, use the HTTPS transport for authenticated requests.

Note: For better security and integrity checking, optionally, you can include the host headers “Content-Type” and “Content-MD5”. Content-MD5 would be:
Base64(HMAC-MD5(UTF-8-Encoding-Of(RequestContentString)))

Mandatory Request Headers

- **Authorization**: it must be composed by three elements, in this format: PortalID:Username:Signature
    - *PortalID*: the Username’s PortalID
    - *Username*: the DNN Username
    - *Signature*: the computed hash; based on the PortalID, Username, the User’s Password, the request verb, the request URI and the included request headers.
- **x-izm-date**: this is the time stamp, as described above

Optional Request Headers

- **X-HTTP-Method-Override**: In case the Web Site is not configured to accept a DELETE verb you can use a POST and add the header: "X-HTTP-Method-Override: DELETE".
- **Content-Type** and **Content-MD5**: when present (one or both of them), they should be included in the Signature hash. (If just one of them is specified, it WILL be computed in the Signature).

How to build the Authorization header (including the Signature)

It is important to build the Signature exactly as described in this document because the same process will happen on the server to validate the request. The computed Signature on the client and the server must be exactly the same. Otherwise, the request will be rejected.

Below is pseudo-code that illustrates the construction of the Authorization request header:

```
Authorization header = PortalID + ":" + Username + ":" + Signature;

Signature = Base64( HMAC-SHA1( UTF-8-Encoding-Of( SecretKey, StringToSign ) ) );

SecretKey = PortalID + LowerCase(Username) + Password;

StringToSign = LowerCase(HTTP-Verb) + LowerCase(<HTTP-Request-URI, from the protocol name up to the query string>) +
CanonicalizedSelectedHeaders;

CanonicalizedSelectedHeaders = <described below>
```

CanonicalizedSelectedHeaders Process:	
For all "mandatory" (except "Authorization") headers and "optional" headers as described above:	
1	Convert each HTTP header name to lower-case. For example, 'X-Izm-Date' becomes 'x-izm-date'.
2	Sort the collection of headers lexicographically by header name.
3	Trim any white-space around the colon in the header. For example, the header 'x-izm-date: 2010-04-30T21:15:55' would become ' x-izm-date:2010-04-30T21:15:55'

Note: If you are i.e. performing a DELETE by means of the POST verb and applying a DELETE X-HTTP-Method-Override header, the "HTTP-Verb" to include in the StringtoSign defined above should be the original value ("POST", in this case).

Code Example – Creating the Signature (VB ASP.NET)

```
Dim headers As New StringBuilder()
headers.AppendLine("x-izm-date: 2015-04-30T20:04:02")
Dim url As String = "http://localhost/DesktopModules/Interzoic-SSO/Service.svc/portal/0/users/user1/isonline"
Dim textToHash As String = BuildStringToHash("get", url, headers.ToString())

Dim portalId As String = "0"
Dim SSOUsername As String = "username"
Dim SSOUserPass As String = "pass"
Dim secretKey As String = portalId + SSOUsername + SSOUserPass
Dim signature = ComputeHash(secretKey, textToHash)
```

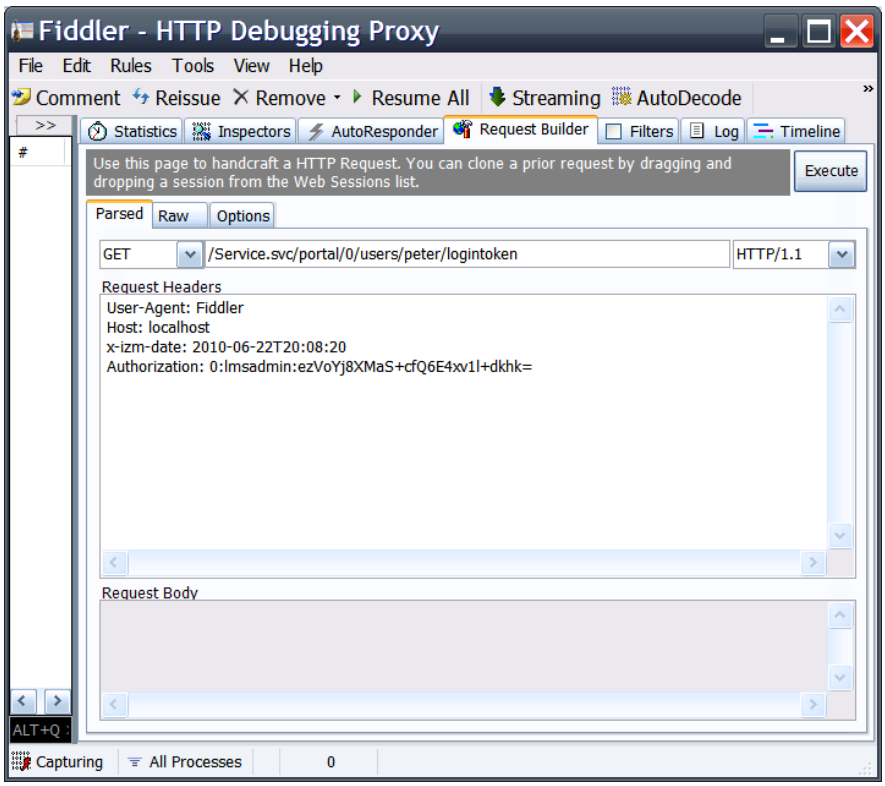
```
Private Shared Function BuildStringToHash(ByVal verb As String, ByVal requestUri As String, ByVal headersList As String) As String
    Dim sb As New StringBuilder()
    sb.Append(verb.ToLower)
    sb.Append(requestUri.ToLower)

    If Not String.IsNullOrEmpty(headersList) Then
        Dim headersArray As String() = headersList.Split(vbCrLf.ToCharArray(), StringSplitOptions.RemoveEmptyEntries)
        Dim contentHeaders As Boolean = False
        Array.Sort(headersArray, StringComparer.OrdinalIgnoreCase)
        For Each head As String In headersArray
            Dim pos As Integer = head.IndexOf(":")
            Dim headerName As String = head.Substring(0, pos).Trim.ToLower
            Dim headerValue As String = head.Substring(pos + 1).Trim
            If headerName.StartsWith("x-izm-") Then
                sb.Append(headerName & ":" & headerValue)
            End If
            If "x-http-method-override,content-type,content-md5".Contains(headerName) Then
                sb.Append(headerName & ":" & headerValue)
            End If
        Next
    End If

    Return sb.ToString()
End Function
```

```
Private Shared Function ComputeHash(ByVal secretKey As String, ByVal stringToHash As String) As String
    Dim secretBytes As Byte() = ASCIIEncoding.ASCII.GetBytes(secretKey)
    Dim hmac As New System.Security.Cryptography.HMACSHA1(secretBytes)
    Dim dataBytes As Byte() = ASCIIEncoding.ASCII.GetBytes(stringToHash)
    Dim computedHash As Byte() = hmac.ComputeHash(dataBytes)
    Dim computedHashString As String = Convert.ToBase64String(computedHash)
    Return computedHashString
End Function
```

Example – Authenticated GET request to get the SSO auto-login token for Username 'peter'.
The request is being authenticated by User 'lmsadmin'



SecretKey = 0lmsadminthepassword

StringToSign = gethttp://www.example.com/desktopmodules/Interzoic-sso/service.svc/portal/0/users/peter/logintokenx-izm-date:2010-06-22T20:08:20

Single Sign-On

Available Services

Token Services

- UserLoginTokenGet (PortalID, Username)

User Services

- UserGet (PortalID, Username)
- UserDelete (PortalID, Username)
- UserRemove (PortalID, Username)
- UserUpdate (PortalID, Username, UserInfoDataContract)
- UserIsOnLine(PortalID, Username)

User Roles Services

- UserRolesGet (PortalID, Username)
- UserRoleGet (PortalID, Username, Rolename)
- UserRoleDrop (PortalID, Username, Rolename)
- UserRoleUpdate (PortalID, Username, Rolename, UserRolesInfoDataContract)

Portal Roles Services

- PortalRolesGet (PortalID)

Profile Services

- UserProfileGet (PortalID, Username)
- UserProfileUpdate (PortalID, Username, ArrayOfUserProfileInfoDataContract)

Note: To simplify the documentation, the examples below do not include any headers.

Token Services

UserLoginTokenGet (PortalID, Username)

Description:
For a given PortalID and Username, returns the temporary User's Login Token.

Workflow

- A remote “Client” application calls the SSO web service hosted on the “DNN/LMS” server to get a temporary Token.
- The Token is valid for a short period of time. The duration is configured in the administrative UI for the web service on the DNN/LMS server.
- The DNN Username AND the Token must be included in the Link/URL from the Client application when directing Users/learners to the DNN/LMS server.
- The DNN User is automatically signed/logged in into the DNN/LMS site if the Token in the URL is still valid.

URI:
GET / portal/{portal}/users/{Username}/logintoken

Return Codes:

200	OK.
403	Forbidden Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
404	Not Found. The User does not exists.
410	Gone. The User has been logically deleted, is currently locked or not approved.

Remarks:
- For security reasons, Host/SuperUser account requests are unsupported. The web service will return 404 regardless if the account is found or not.

Example:
GET <root url>/portal/0/users/peter/logintoken

Returns XML:
<string
xmlns="**http://schemas.microsoft.com/2003/10/Serialization/**">**133033134266066777133033134**</string>

From the remote website, users can be directed to any page on the DNN portal that has the SSO Auto-Login module present. The URL must include the two query string values: **ssouser** and **ssoToken**.

If successful, the User will be redirected to the Success Redirect URL. If rejected, the User will be redirected to the Reject Redirect URL. Both of these URLs are set in the SSO Configuration page.

This allows a single page in the site to hold the “SSO Auto-Login module” (it can even be a hidden page) and then redirect to whatever target page you want depending on success or reject.

Optionally, you can also include a **successUrl** to redirect to another page if the login is successfully completed and/or a **rejectUrl** to redirect to another page if the login is not successfully completed. The query string redirect URLs have priority over the default redirect URL settings on the SSO Configuration page.

Note: Only relative path URLs are supported in the query string successUrl or rejectUrl name/value pairs, ex. rejectUrl=/Login/Default.aspx. This is to avoid cross-site request forgery attempts.

Example:

https://www.example.com/Hidden/SSO.aspx
?**ssouser**=peter&**ssotoken**=133033134266066777133033134&
successurl=/MyCourses.aspx&**rejecturl**=/LoginError.aspx

If the auto-login is successful, the User will be automatically redirected to /MyCourses.aspx. If the auto-login fails, the User will be automatically redirected to /LoginError.aspx.

If redirect URLs are not provided in either the query string or the SSO Configuration the User will simply remain on the Landing page - in this Example: /Hidden/SSO.aspx. If successful, the module would remain hidden without presenting anything. If rejected and the Enable SSO Failure Message is enabled an error message would be displayed: either "Login rejected for the following reason: Expired Token" or "Login rejected for the following reason: Invalid User / Token Pair".

User Services

UserGet (PortalID, Username)

Description:
For a given PortalID and Username, returns the User's Account.

URI:
GET /portal/{PortalID}/users/{Username}

Return Codes:

200	OK.
403	Forbidden Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
404	Not Found. User does not exists.
410	Gone. User deleted, currently locked or not approved.

Remarks:
- For security reasons, Host/SuperUser accounts will NOT be returned. It will return 404 as if the User does not exist.

Example:
GET <root url>/portal/0/users/admin

Returns XML:

```
<UserInfoDataContract xmlns="http://schemas.datacontract.org/2004/07/Interzoic_SSO.Shared"
      xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <DisplayName>Administrator Account</DisplayName>
  <Email>admin@change.me</Email>
  <FirstName>Administrator</FirstName>
  <IsSuperUser>false</IsSuperUser>
  <LastName>Account</LastName>
  <Password> the@adminpassw0rd</Password>
  <PortalID>0</PortalID>
<Roles xmlns:a="http://schemas.microsoft.com/2003/10/Serialization/Arrays">
  <a:string>Registered Users</a:string>
```

```
 <a:string>Subscribers</a:string>
 <a:string>Administrators</a:string>
 <a:string>uno uno</a:string>
    </Roles>
 <UserID>2</UserID>
 <Username>admin</Username>
</UserInfoDataContract>
```

UserDelete (PortalID, Username)

Description:
For a given PortalID and Username, logically deletes the User's Account.

URI:
DELETE /portal/{PortalID}/users/{Username}

Return Codes:

200	OK.
403	Forbidden Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
404	Not Found. User does not exist.
410	Gone. User already deleted.

Remarks:

- For security reasons, Host/SuperUser accounts will NOT be deleted. HTTP status code 404 will be returned.

- The main portal's administrator User cannot be deleted. HTTP status code 500 will be returned, same as when a 'valid' User delete operation fails for some reason.

- In case the Web Site is not configured to accept a DELETE verb (typically if this is the case, you would get a "403 Forbidden" HTTP return code; or it could be a different error code) you can use a POST and add the header: "X-HTTP-Method-Override: DELETE".

Example:
DELETE <root url>/portal/0/users/someUsername

Returns: HTTP status code 200 (OK)

UserRemove (PortalID, Username)

Description:
For a given PortalID and Username, physically deletes the User's Account.

URI:
DELETE /portal/{PortalID}/users/{Username}/remove

Return Codes:

200	OK.
403	Forbidden Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
404	Not Found. User does not exist.

Remarks:

- For security reasons, Host/SuperUser accounts will NOT be deleted. HTTP status code 404 will be returned.

- The main portal's administrator User cannot be deleted. HTTP status code 500 will be returned, same as when a 'valid' User delete operation fails for some reason.

- In case the Web Site is not configured to accept a DELETE verb (typically if this is the case, you would get a "403 Forbidden" HTTP return code; or it could be a different error code) you can use a POST and add the header: "X-HTTP-Method-Override: DELETE".

Example:
DELETE <root url>/portal/0/users/someUsername/remove

Returns: HTTP status code 200 (OK)

UserUpdate (PortalID, Username, UserInfoDataContract)

Description:
For a given PortalID, UserID and UserInfoDataContract, creates or updates the User's Account.

UserInfoDataContract = same XML structure as described for the "UserGet" method. It includes the User's basic information, like first, last, display name, email and Password.

URI:
POST /portal/{PortalID}/users/{Username}

Return Codes:

200	OK.
403	Forbidden Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
410	Gone. User deleted.

Remarks:

- For security reasons, Host/SuperUser accounts cannot be added or edited. It will return 400 – Bad Request.

- For security reasons, you are not allowed to change a 'IsSuperUser' flag. It will return 400 – Bad Request.

- For a new User, "<Roles>" if specified, will be ignored. Every new User will have just the default 'automatic' (auto assigned) security roles. By default in DNN, they are "Registered Users" and "Subscribers". In your specific DNN site, they will be all the roles with this setting enabled.

- For security reasons, you are not allowed to update a User's "Username". In fact, if you change the Username and perform a new UserUpdate, you will be creating this new "Username" (in case it is new).

- The specified {Username} in the URI and in the posted data must match. Otherwise you will get error code 400 (bad request).

- As defined in the current DNN portal authentication policy, a weak or blank Password will be rejected (400 return code, bad request).

- The PortalID in the posted "UserInfoDataContract" is ignored. What matters is the PortalID specified in the URI

- There is no need to include all the data in the UserInfoDataContract. Some fields can be missing. i.e. <Roles>, <IsSuperUser> , <UserId>, <PortalID>; which would be ignored when creating a User.

- When creating a new User, if <DisplayName> is not explicitly provided, it will be set as FirstName + LastName

- When creating a new User, you must specify the Password (it will not be auto-generated).

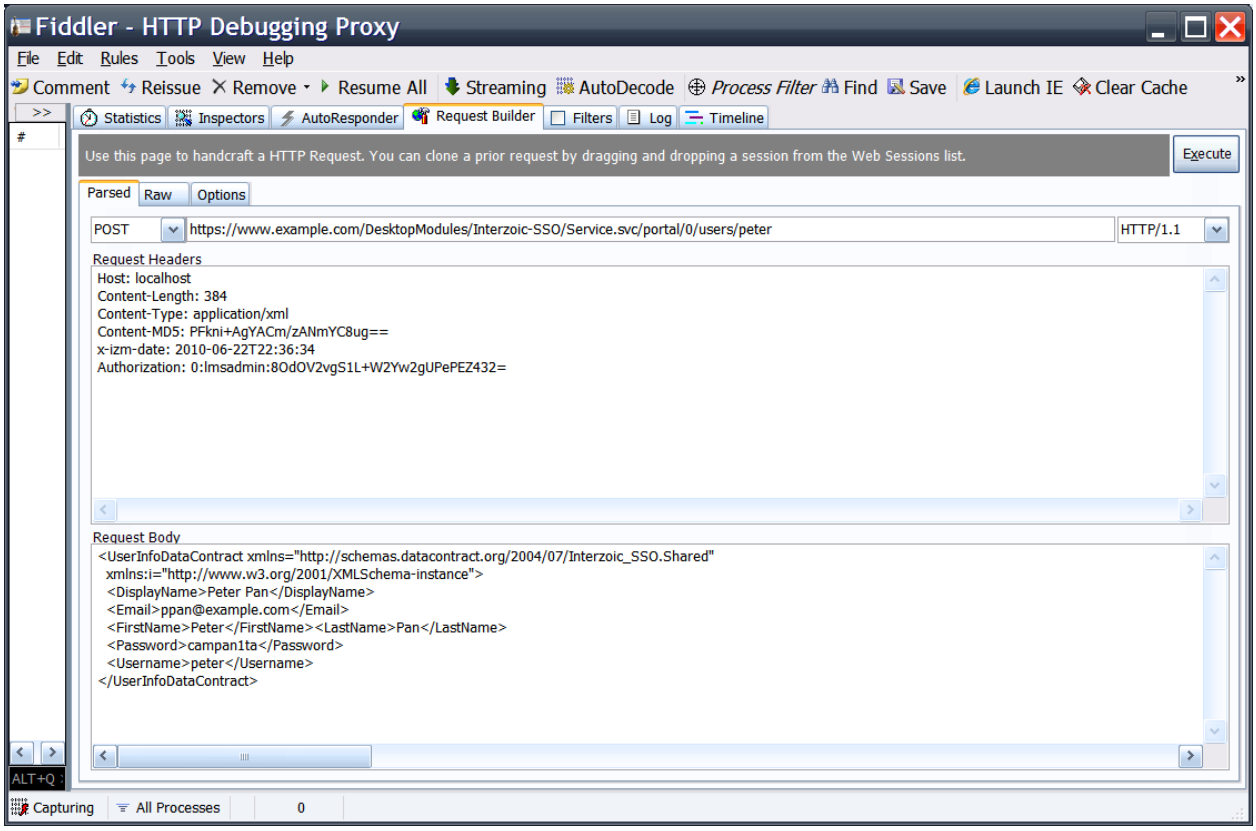
- When creating or updating a User, if already exists a User with the same 'Username', but flagged as deleted, the operation is rejected (410 HTTP return code).

- For new or updated User's information, no email notification will be sent.

Example:
The example below will create a new User 'peter'.
We are using Fiddler to show the actual POST request sent to the service. This provided a clear 'platform independent' working Example for this web service call. This example includes the authorization headers.

How to programmatically build this request depends on each client application/programming tool.

POST <root url>/portal/0/users/peter



UserIsOnLine (PortalID, Username)

Description:
For a given PortalID and Username, returns an integer specifying the current on-line status for the user.

Possible return values are:

- -1 : "UsersOnline" feature is disabled on the DNN instance (Host > Host Settings > Advanced Settings > Other Settings > Enable Users Online?)
- 1 : The user is on-line (logged in)
- 0 : The user is off-line (not logged in)

Required Web site configuration for this feature to work
Host > Host Settings > Advanced Settings > Other Settings > Enable Users Online?

Relevant web.config setting
<membership defaultProvider="AspNetSqlMembershipProvider" **userIsOnlineTimeWindow="15"**>

userIsOnlineTimeWindow = how long a user is assumed still to be online after his last activity.
IMPORTANT: when the user 'Logout', will be still seen active until the 'userIsOnlineTimeWindow' timeouts.

URI:
GET / portal/{portal}/users/{Username}/ isonline

Return Codes:

200	OK.
403	Forbidden Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
404	Not Found. The User does not exists.
410	Gone. The User has been logically deleted, is currently locked or not approved.

Remarks:
- For security reasons, Host/SuperUser account requests are unsupported. The web service will return 404 regardless if the account is found or not.

Example:
GET <root url>/portal/0/users/peter/ isonline

Returns XML:
<int xmlns="**http://schemas.microsoft.com/2003/10/Serialization/**">**1**</int>

User Roles Services

UserRolesGet (PortalID, Username)

Description:
For a given PortalID and Username, returns the User's Roles.

URI:
GET /portal/{PortalID}/users/{Username}/roles

Return Codes:

200	OK.
403	Forbidden Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
404	Not Found.
410	Gone. User deleted.

Remarks:
- Host/SuperUser accounts will be rejected as ‘ not found’ Users (404 HTTP return code)

Example:
GET <root url>/portal/0/users/peter/roles

Returns XML:

```
<ArrayOfUserRolesInfoDataContract
      xmlns="http://schemas.datacontract.org/2004/07/Interzoic_SSO.Shared"
      xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
<UserRolesInfoDataContract>
 <EffectiveDate>0001-01-01T00:00:00</EffectiveDate>
 <ExpiryDate>0001-01-01T00:00:00</ExpiryDate>
 <RoleGroupID>-1</RoleGroupID>
 <RoleName>Registered Users</RoleName>
    </UserRolesInfoDataContract>
<UserRolesInfoDataContract>
 <EffectiveDate>0001-01-01T00:00:00</EffectiveDate>
 <ExpiryDate>0001-01-01T00:00:00</ExpiryDate>
 <RoleGroupID>-1</RoleGroupID>
 <RoleName>Subscribers</RoleName>
    </UserRolesInfoDataContract>
<UserRolesInfoDataContract>
 <EffectiveDate>0001-01-01T00:00:00</EffectiveDate>
 <ExpiryDate>0001-01-01T00:00:00</ExpiryDate>
 <RoleGroupID>1</RoleGroupID>
 <RoleName>some role</RoleName>
    </UserRolesInfoDataContract>
<UserRolesInfoDataContract>
 <EffectiveDate>2010-04-30T00:00:00</EffectiveDate>
 <ExpiryDate>2010-05-10T00:00:00</ExpiryDate>
 <RoleGroupID>0</RoleGroupID>
 <RoleName>other role</RoleName>
    </UserRolesInfoDataContract>
  </ArrayOfUserRolesInfoDataContract>
```

UserRoleDrop (PortalID, Username, Rolename)

Description:
For a given PortalID, Username and Rolename, drops the Role from the User.

URI:
DELETE /portal/{PortalID}/users/{Username}/roles/{rolename}

Return Codes:

200	OK.
403	Forbidden Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
404	Not Found. Role or User-Role does not exists.
410	Gone. User deleted.

Remarks:

- For security reasons, Host/SuperUser accounts (which do not have any role) will return HTTP status code 404 (not found) same as a true ‘not found’ User.

- In case the Web Site is not configured to accept a DELETE verb (typically if this is the case, you would get a "403 Forbidden" HTTP return code; or it could be a different error code) you can use a POST and add the header: “X-HTTP-Method-Override: DELETE”.

UserRoleGet (PortalID, Username, Rolename)

Description:
For a given PortalID, Username and Rolename, returns the user role's information.

URI:
GET /portal/{PortalID}/users/{Username}/roles/{rolename}

Return Codes:

200	OK.
403	Forbidden Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
404	Not Found.
410	Gone. User deleted.

Remarks:
- Host/SuperUser accounts will be rejected as ' not found' Users (404 HTTP return code)

Example:
GET <root url>/portal/0/users/peter/role/registered%20users

Returns XML:

```
<UserRoleInfoDataContract xmlns="http://schemas.datacontract.org/2004/07/Interzoic_SSO.Shared"
      xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    <EffectiveDate>2010-04-30T00:00:00</EffectiveDate>
    <ExpiryDate>2011-05-10T00:00:00</ExpiryDate>
    <RoleGroupID>-1</RoleGroupID>
    <RoleName>Registered Users</RoleName>
 </UserRoleInfoDataContract>
```

UserRoleUpdate (PortalID, Username, Rolename, UserRolesInfoDataContract)

Description:
For a given PortalID, Username, Rolename and UserRolesInfoDataContract, assigns the Role to the User. If the role was already assigned, the assign information (i.e. the effective and expiry date) is updated.

UserRolesInfoDataContract = same XML structure as described within the ArrayOfUserRolesInfoDataContract XML structure above.

Sample XML:

```
<UserRoleInfoDataContract xmlns="http://schemas.datacontract.org/2004/07/Interzoic_SSO.Shared"
      xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    <EffectiveDate>2010-04-30T00:00:00</EffectiveDate>
    <ExpiryDate>2011-05-10T00:00:00</ExpiryDate>
    <RoleGroupID>-1</RoleGroupID>
    <RoleName>Registered Users</RoleName>
 </UserRoleInfoDataContract>
```

URI:
POST /portal/{PortalID}/users/{Username}/roles/{rolename}

Return Codes:

200	OK.
403	Forbidden Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
404	Not Found. User does not exists.
410	Gone. User deleted.

Remarks:

- For security reasons, Host/SuperUser accounts (which do not have any role) will return HTTP status code 404 (not found) same as a true 'not found' User.

- EffectiveDate and ExpiryDate can be "null", which means "no limit"

- RoleGroupID "-1" is the built-in DNN group of roles "<Global Roles>"

- No email notification is sent.

- Regarding the posted "UserRolesInfoDataContract" data; when adding or updating a User's role information, RoleGroupID can be omitted or can have whatever value, even 'null'. It will be always ignored.
  However, the RoleName must match the one specified in the URI. Otherwise you'll get a 400 return code (bad request).

Portal Roles Services

PortalRolesGet (PortalID)

Description:
For a given PortalID, returns the Portal's Roles.

URI:
GET /portal/{PortalID}/roles

Return Codes:

200	OK.
403	Forbidden Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.

Example:

GET <root url>/portal/0/roles

Returns XML:

```
<ArrayOfPortalRoleInfoDataContract
      xmlns="http://schemas.datacontract.org/2004/07/Interzoic_SSO.Shared"
      xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
<PortalRoleInfoDataContract>
  <Description>Portal Administration</Description>
  <RSVPCode />
  <RoleGroupID>-1</RoleGroupID>
  <RoleID>0</RoleID>
  <RoleName>Administrators</RoleName>
</PortalRoleInfoDataContract>
<PortalRoleInfoDataContract>
  <Description />
  <RSVPCode>AAA</RSVPCode>
  <RoleGroupID>1</RoleGroupID>
  <RoleID>9</RoleID>
  <RoleName>Company A</RoleName>
</PortalRoleInfoDataContract>
<PortalRoleInfoDataContract>
  <Description>Registered Users</Description>
  <RSVPCode />
  <RoleGroupID>-1</RoleGroupID>
  <RoleID>1</RoleID>
  <RoleName>Registered Users</RoleName>
</PortalRoleInfoDataContract>
<PortalRoleInfoDataContract>
  <Description>A public role for portal subscriptions</Description>
  <RSVPCode />
  <RoleGroupID>-1</RoleGroupID>
  <RoleID>2</RoleID>
  <RoleName>Subscribers</RoleName>
</PortalRoleInfoDataContract>
<PortalRoleInfoDataContract>
  <Description>A social role </Description>
  <RSVPCode />
  <RoleGroupID>-1</RoleGroupID>
  <RoleID>9</RoleID>
  <RoleName>Social Role</RoleName>
  <RoleType>None</RoleType>
  <SecurityMode>SocialGroup</SecurityMode>
  <Status>Approved</Status>
</PortalRoleInfoDataContract>
</ArrayOfPortalRoleInfoDataContract>
```

Profile Services

UserProfileGet (PortalID, Username)

Description:
For a given PortalID and Username, returns the User's Profile.

URI:
GET /portal/{PortalID}/users/{Username}/profile

Return Codes:

200	OK.
403	Forbidden

	Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
404	Not Found.
410	Gone. User deleted.

Remarks:

- Host/SuperUser accounts will be rejected as ‘ not found’ Users (404 HTTP return code)

Example:

GET <root url>/portal/0/users/peter/profile

Returns XML:

```
<ArrayOfUserProfileInfoDataContract
      xmlns="http://schemas.datacontract.org/2004/07/Interzoic_SSO.Shared"
      xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
<UserProfileInfoDataContract>
<PropertyName>Prefix</PropertyName>
<PropertyValue i:nil="true" />
   </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
<PropertyName>FirstName</PropertyName>
<PropertyValue>Peter</PropertyValue>
   </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
<PropertyName>MiddleName</PropertyName>
<PropertyValue i:nil="true" />
   </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
<PropertyName>LastName</PropertyName>
<PropertyValue>Pan</PropertyValue>
   </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
<PropertyName>Suffix</PropertyName>
<PropertyValue>Jr</PropertyValue>
   </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
<PropertyName>Unit</PropertyName>
<PropertyValue i:nil="true" />
   </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
<PropertyName>Street</PropertyName>
<PropertyValue i:nil="true" />
   </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
<PropertyName>City</PropertyName>
<PropertyValue i:nil="true" />
   </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
<PropertyName>Region</PropertyName>
<PropertyValue i:nil="true" />
   </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
```

```
 <PropertyName>Country</PropertyName>
 <PropertyValue>Canada</PropertyValue>
    </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
 <PropertyName>PostalCode</PropertyName>
 <PropertyValue>14210</PropertyValue>
    </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
 <PropertyName>Telephone</PropertyName>
 <PropertyValue i:nil="true" />
    </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
 <PropertyName>Cell</PropertyName>
 <PropertyValue i:nil="true" />
    </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
 <PropertyName>Fax</PropertyName>
 <PropertyValue i:nil="true" />
    </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
 <PropertyName>Website</PropertyName>
 <PropertyValue>www.example.com</PropertyValue>
    </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
 <PropertyName>IM</PropertyName>
 <PropertyValue i:nil="true" />
    </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
 <PropertyName>Biography</PropertyName>
 <PropertyValue><h2>Bio</h2>this is the<br />bio of Peter</PropertyValue>
    </ UserProfileInfoDataContract >
<UserProfileInfoDataContract>
 <PropertyName>TimeZone</PropertyName>
 <PropertyValue>2</PropertyValue>
    </UserProfileInfoDataContract>
<UserProfileInfoDataContract>
 <PropertyName>PreferredLocale</PropertyName>
 <PropertyValue>en-US</PropertyValue>
    </ UserProfileInfoDataContract >
    </ArrayOfUserProfileInfoDataContract>
```

UserProfileUpdate (PortalID, Username, ArrayOfUserProfileInfoDataContract)

Description:
For a given PortalID, Username and an ArrayOfUserProfileInfoDataContract, updates the User's Profile.

ArrayOfUserProfileInfoDataContract = same XML structure described above.

URI:
POST /portal/{PortalID}/users/{Username}/profile

Return Codes:

200	OK.
403	Forbidden

	Authentication and/or authorization failed.
400	Bad Request. The document in the entity-body, if any, is an error message.
500	Internal Server Error. The document in the entity-body, if any, is an error message.
404	Not Found. User does not exists.
410	Gone. User deleted.

Remarks:

- For security reasons, Host/SuperUser accounts (which do not have any role) will return HTTP status code 404 (not found) same as a true 'not found' User.

- In the posted UserProfileInfoDataContract, you can include or many/all "properties", as long as they are defined in the DNN Portal and you provide valid/expected values for each one.

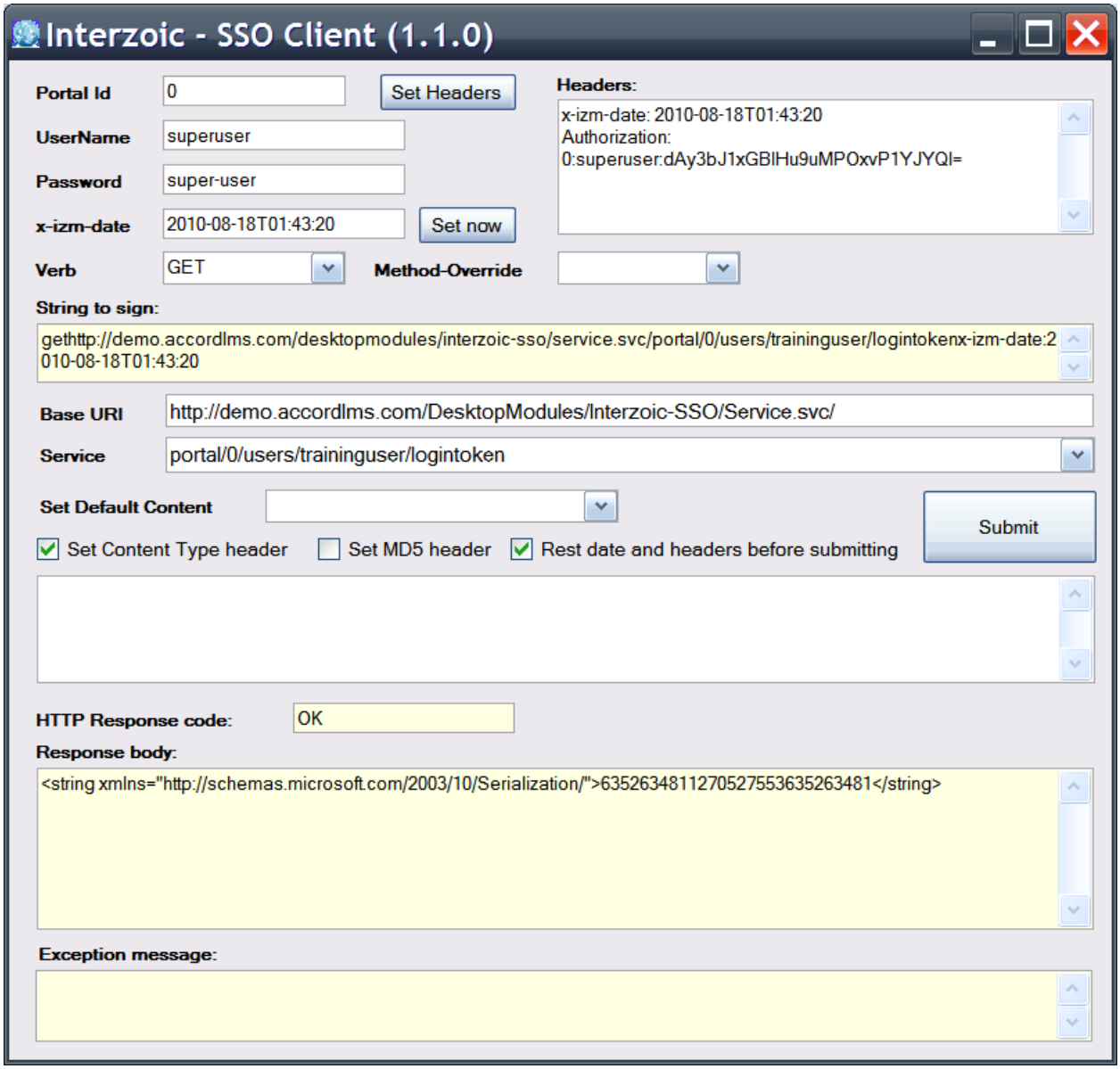
- Invalid/unknown properties will be ignored (no error will be returned and the POST will not be rejected. Valid profile properties will be set and the invalid ones will be ignored.

SSO Client - Testing Harness

A .NET Windows client testing harness, SSOClient.exe, is included at no additional cost. It exercises the full range of SS0 web services. Both the executable and all the VB.net source code are provided. This makes it easy to create your own communications in any programming language and provides best practices working examples if you will be communicating from another platform.

Auto-Login: Sample usage

Use the provided Testing Harness to get a login token for the “trainingUser” account.

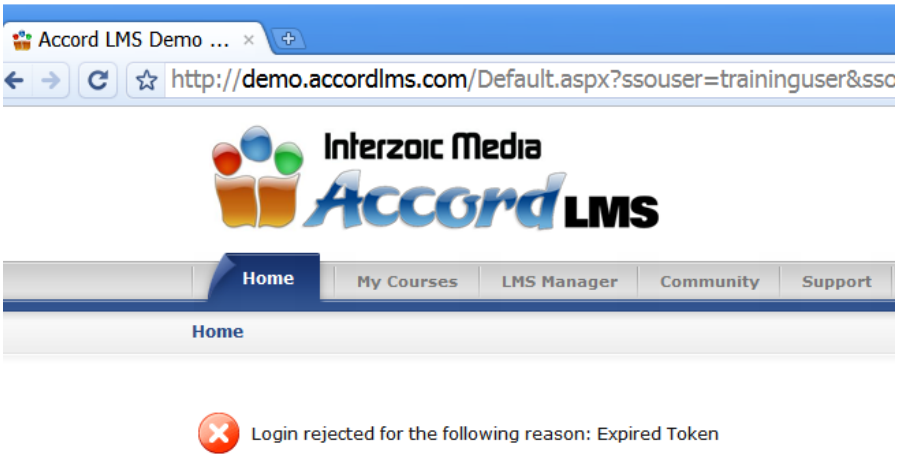


If you replace the token you just created with the token in the example URL below, you can automatically login the “trainingUser” User to the Accord LMS demo site.

http://demo.accordlms.com/Default.aspx?ssouser=trainingUser&ssotoken=11704568232340913859117 0456823

If the auto-login was accepted, you will be redirected to http://demo.accordlms.com/MyCourses.aspx, as configured in the module's Configuration page.

If you browse to this URL after the configured "Token Timeout" expires (90 seconds in this sample) you will get an error message:



You will see this message because no RejectURL was set on the configuration page and the Enable SSO Failure Message was enabled.

Also, as defined in the Configuration page, the web services requests are being logged into the DNN Event Log:

7/6/2010 7:16:43 PM Admin Alert **Interzoic-SSO**: ACCEPTED superUser: GET http://demo.accordlms.com/des ...

Interzoic-SSO: ACCEPTED superUser: GET https://demo.accordlms.com/desktopmodules/interzoic-sso/service.svc/portal/0/Users/trainingUser/logintoken
Portal: 0
User: superUser
Verb: GET
URI: https://demo.accordlms.com/desktopmodules/interzoic-sso/service.svc/portal/0/Users/trainingUser/logintoken

7/6/2010 7:33:46 PM Admin Alert **Interzoic-SSO**: REJECTED superUser: GET http://demo.accordlms.com/des ...

Interzoic-SSO: REJECTED superUser: GET https://demo.accordlms.com/desktopmodules/interzoic-sso/service.svc/portal/0/Users/trainingUser/logintoken
Portal: 0
User: superUser
Verb: GET
URI: https://demo.accordlms.com/desktopmodules/interzoic-sso/service.svc/portal/0/Users/trainingUser/logintoken
Reject Code: RequestTimeTooSkewed
Reject Message: The request has expired (invalid date-time set in the request header). Server time: 2010-07-06T23:33:46 Request time: 2010-07-06T22:27:04

By means of the provided SSOClient.exe testing harness you can exercise all the provided web services available on the 'help' page, which are also described in the sections above.

SSO Client - .NET API library

A .NET client clibrary is included at no additional cost. It exercises the full range of SS0 web services. Both the DLL and all the VB.net source code are provided. This makes it easy to create your own communications in any .NET language and provides best practices working examples if you will be communicating from another platform.

Sample usages:

Getting a login token

```
Dim credentials As New Interzoic.SSO.Client.API.Credentials(0,
"AuthenticateUserName", "UserPassword")

Dim baseUri As String = "http://www.example.com/DesktopModules/Interzoic-
SSO/Service.svc/"

Dim client As New Interzoic.SSO.Client.API.Connection(credentials, baseUri)

Dim statusCode As System.Net.HttpStatusCode

Dim token As String = client.GetUserLoginToken(0, "UserName", statusCode)
```

Getting the user's roles

```
Dim credentials As New Interzoic.SSO.Client.API.Credentials(0,
"AuthenticateUserName", "UserPassword")

Dim baseUri As String = "http://www.example.com/DesktopModules/Interzoic-
SSO/Service.svc/"

Dim client As New Interzoic.SSO.Client.API.Connection(credentials, baseUri)

Dim statusCode As System.Net.HttpStatusCode

Dim userRolesList As List(Of UserRoleInfoDataContract) = client.UserRolesGet(0,
"UserName", statusCode)
```