User Provisioning Client API v. 3.0.1 Documentation

Contents

[Introduction 1](#_Toc493593307)

[Usage 1](#_Toc493593308)

[Example 1](#_Toc493593309)

[Class Documentation 4](#_Toc493593310)

[UserProvHelper 4](#_Toc493593311)

[Constructors 4](#_Toc493593312)

[Public Methods 4](#_Toc493593313)

[User Class 6](#_Toc493593314)

[Public Properties 6](#_Toc493593315)

[Security Class 6](#_Toc493593316)

[Role Class 7](#_Toc493593317)

[Resource Class 7](#_Toc493593318)

[Permission Class 7](#_Toc493593319)

[Application 8](#_Toc493593320)

[Architecture 8](#_Toc493593321)

# Introduction

The User Provisioning Client API is a set of tools that handles communicating with the User Provisioning Web application to get information regarding the security on the application. It does this through use of the HttpClient library in C#. Essentially this library calls out to the User Provisioning Web Interface and receives JSON from it and translates that into the internal Data transfer objects.

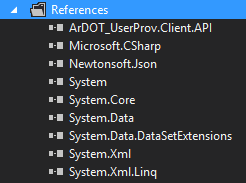
# Usage

To use the User Provisioning Client API you must either include the source code in your project or include the ArDOT\_UserProv.Client.API.dll file and the Newtornsoft.Json.dll file. The next step is to import ArDOT\_UserProv.Client.API. The UserProvHelper class does all of the work. The other classes are just Models.

## Example

The Following example uses the ArDOT\_UserProv.Client.API UserProvHelper with the UserProvUrl and UserProvApplicationId appsettings in the App.Config. The example calls the GetApplicationSecurities method and prints the output table. The example has quite a few lines of code handling errors to give good output messages.

#### References



#### App.config

<?xml version="1.0" encoding="utf-8"?>

<configuration>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />

</startup>

<appSettings>

<add key="UserProvUrl" value="http://290-app-cf/UserProv/" />

<add key="UserProvApplicationId" value="USER\_PROV" />

<add key="ClientSettingsProvider.ServiceUri" value="" />

</appSettings>

<system.web>

<membership defaultProvider="ClientAuthenticationMembershipProvider">

<providers>

<add name="ClientAuthenticationMembershipProvider" type="System.Web.ClientServices.Providers.ClientFormsAuthenticationMembershipProvider, System.Web.Extensions, Version=4.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" serviceUri="" />

</providers>

</membership>

<roleManager defaultProvider="ClientRoleProvider" enabled="true">

<providers>

<add name="ClientRoleProvider" type="System.Web.ClientServices.Providers.ClientRoleProvider, System.Web.Extensions, Version=4.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" serviceUri="" cacheTimeout="86400" />

</providers>

</roleManager>

</system.web>

</configuration>

#### Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Configuration;

using System.Net.Http;

using ArDOT\_UserProv.Client.API;

namespace ArDOT\_UserProv.Client {

class Program {

static void Main(string[] args) {

UserProvHelper UserProvInterface = null;

try {

UserProvInterface = new UserProvHelper();

}

catch (Exception e) {

Exception loopVar = e;

while (loopVar != null) {

Console.WriteLine(loopVar.Message);

loopVar = loopVar.InnerException;

}

Console.WriteLine(e.StackTrace);

}

try {

Console.WriteLine("GetApplicationSecurities ApplicationId:" + UserProvInterface.GetUserProvApplicationId());

Security[] ApplicationUsers = UserProvInterface.GetApplicationSecurities();

string tableFormat = "| {0,13} | {1,11} | {2,13} | {3,13} | {4,13} |";

string headerRow = String.Format(tableFormat, "App Id", "User Id", "Role Id", "Perm Id", "Res Id");

string seperateRow = "";

for (int i = 0; i < headerRow.Length; i++) {

seperateRow += "=";

}

Console.WriteLine(seperateRow);

Console.WriteLine(headerRow);

Console.WriteLine(seperateRow);

if (ApplicationUsers.Length > 0) {

foreach (Security sec in ApplicationUsers) {

Console.WriteLine(tableFormat,

sec.Application.ApplicationId,

sec.User.UserName,

sec.Role == null ? "" : sec.Role.RoleId,

sec.Permission == null ? "" : sec.Permission.PermissionId,

sec.Resource == null ? "" : sec.Resource.ResourceId

);

}

}

else {

Console.WriteLine("| {0,-75} |", "No Records Found...");

}

Console.WriteLine(seperateRow);

}

catch (Exception e) {

Exception loopVar = e;

while (loopVar != null) {

Console.WriteLine(loopVar.Message);

loopVar = loopVar.InnerException;

}

Console.WriteLine(e.StackTrace);

}

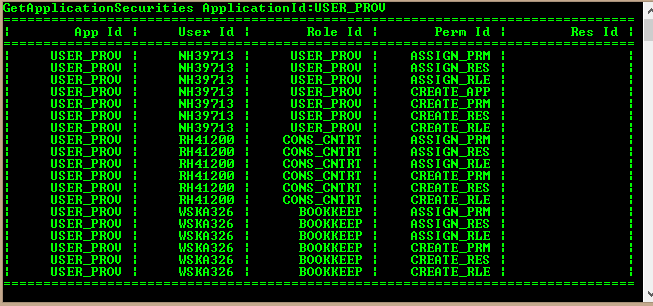
Console.ReadLine();

}

}

}

#### Output



# Class Documentation

## UserProvHelper

Written By: Nathan Hill (NH39713)

Date: 09/20/2017

The User Prov Helper class is designed to handle the complexity of communincating with the webserver for the information. It is to provide a clean concise interface to the User Provisioning security data for applications to get their security. It can use the UserProvUrl and UserProvApplicationId application settings in the config file with the default constructor or they can be specified in the constructor.

### Constructors

#### Default

The Default constructor uses the UserProvUrl and UserProvApplicationId settings in the App config file. This makes the Move to production easier because the setting can be changed in the config just like the connection strings.

#### Parameterized

The parameterized constructor uses the passed parameters to connect to the Server. This will make move to production harder but it allows you to specify the ApplicationId programmatically. This could be useful in the case that you need to know security for two different applications.

### Public Methods

UserProvHelper Has 14 public methods 12 correspond to actions on the web api on the User Prov web system.

#### GetApplicationSecurities

public Security[] GetApplicationSecurities()

The GetApplicationSecurities method returns all of the securities for the application. NOTE: This is returning all data. So if the Application has a high number of users this could take longer. This should only be used if nothing else will work.

#### GetApplicationSecuritiesForUser

public Security[] GetApplicationSecuritiesForUser(string UserId)

The GetApplicationSecuritiesForUser method returns the Securities for the given UserId.

#### GetApplictionRoles

public Role[] GetApplictionRoles()

The GetApplicationRoles method returns the Roles for the Application. This is just a Role table dump filtering on the ApplicationId.

#### GetApplicationPermissions

public Permission[] GetApplicationPermissions()

The GetApplicaitonPermissions method returns the Permissions for the Application. This is just a Permission table dump filtering on the ApplicationId.

#### GetApplicationResources

public Resource[] GetApplicationResources()

The GetApplicationResource method returns the Resources for the Application. This is just a Resource table dump filtering on the ApplicationId.

#### GetApplicationUsersInRole

public User[] GetApplicationUsersInRole(string RoleId)

The GetApplicationUsersInRole method returns the Users that are in the given role.

#### GetApplicationUsersInPermission

public User[] GetApplicationUsersInPermission(string PermissionId)

The GetApplicationUsersInPermission method returns the Users that are in the given permission.

#### GetApplicationUsersInResource

public User[] GetApplicationUsersInResource(string ResourceId)

The GetApplicationUsersInResource method returns the Users that are in the given resource.

#### GetApplicaitonUserRoles

public Role[] GetApplicaitonUserRoles(string UserId)

The GetApplicationUserRoles method returns the Roles that the given User has. NOTE: The GetApplicationSecuritiesForUser method returns this information as well as the Permissions and Resources.

#### GetApplicationUserPermissions

public Permission[] GetApplicationUserPermissions(string UserId)

The GetApplicationUserPermissions method returns the Permissions that the given User has. NOTE: The GetApplicationSecuritiesForUser method returns this information as well as the Roles and Resources.

#### GetApplicationUserResources

public Resource[] GetApplicationUserResources(string UserId)

The GetApplicationUserResources method returns the Resources that the given User has. NOTE: The GetApplicationSecuritiesForUser method returns this information as well as the Roles and Permissions.

#### GetUserProvUrl

public string GetUserProvUrl()

#### Get the URL that the UserProvHelper is using to connect to the UserProv System.

#### GetUserProvApplicationId

public string GetUserProvApplicationId()

Get the ApplicationId the UserProvHelper is using to connect to UserProv System.

## User Class

The User class is a data transfer object used to carry information about a given user.

### Public Properties

The User class uses properties to hold the User data.

#### UserName

public string UserName { get; set; }

The UserName property contains the Users Windows Account name also called UserId.

#### EmployeeNumber

public string EmployeeNumber { get; set; }

The EmployeeNumber contains the Employee Number for the User. NOTE: This may not be set in the case of a contractor.

#### FirstName

public string FirstName { get; set; }

The FirstName contains the First Name for the User.

#### LastName

public string LastName { get; set; }

The LastName contains the Last Name for the User.

#### Budget

public string Budget { get; set; }

The Budget contains the 3 digit budget number that the User is in.

#### BudgetName

public string BudgetName { get; set; }

The BudgetName contains the Name of the budget that the User is in.

## Security Class

The Security class is a data transfer object used to carry information about a relationship between a User, Application, Role, Permission, and Resource. NOTE: Both the Legacy and Normal Security Models are flattened here.

#### User

public User User { get; set; }

The User property contains the User that the Security Instance applies too.

#### Application

public Application Application { get; set; }

The Application property contains the Application that the Security Instance applies too.

#### Role

public Role Role { get; set; }

The Role property contains the Role that the Security Instance is granting.

#### Permission

public Permission Permission { get; set; }

The Permission property contains the Permission that the Security Instance is granting.

#### Resource

public Resource Resource { get; set; }

The Resource property contains the Resource that the Security Instance is granting.

## Role Class

The Role class is a data transfer object used to carry information about a role.

#### RoleId

public string RoleId { get; set; }

The RoleId property contains the Id used by the system to identify the Role.

#### RoleName

public string RoleName { get; set; }

The RoleName property contains the Name for the Role.

#### RoleDescription

public string RoleDescription { get; set; }

The RoleDescription property contains the Description for the Role.

## Resource Class

The Resource class is a data transfer object used to carry information about the Resource.

#### ResourceId

public string ResourceId { get; set; }

The ResourceId property contains the Id used by the system to identify the Resource.

#### ResourceName

public string ResourceName { get; set; }

The ResourceName property contains the Name for the Resource.

#### ResourceDescription

public string ResourceDescription { get; set; }

The ResourceDescription property contains the Description for the Resource.

## Permission Class

The Permission class is a data transfer object used to carry information about the Permission.

#### PermissionId

public string PermissionId { get; set; }

The PermissionId property contains the Id that the system uses to identify the Permisison.

#### PermissionName

public string PermissionName { get; set; }

The PermissionName property contains the Name of the Permission.

#### PermissionDescription

public string PermissionDescription { get; set; }

The PermissionDescription property contains the Description of the Permission.

## Application

The Application class is a data transfer object used to carry information about an Application.

#### ApplicationId

public string ApplicationId { get; set; }

The ApplicationId property contains the Id that the system uses to identify the Application.

#### ApplicationName

public string ApplicationName { get; set; }

The ApplicationName property contains the Name of the Application.

#### ApplicationDescription

public string ApplicationDescription { get; set; }

The ApplicationDescription property contains the Description of the Application.

# Architecture

The ArDOT\_UserProv.Client.API library is built from 7 C# files. Each class has a file named like the class. The library has references to System.Configuration, System.Net.Http, and System.Web.Extensions libraries. All of the logic is held in UserProvHelper. The other classes are just Data Transfer Objects.