DMCM .NET Library

Version 0.2

October 11, 2013

Purpose

The Dell Multi Cloud Manager (DMCM, formerly Enstratius) .NET library is an early-stage, proof of concept project that allows .NET applications to consume a fraction of the DMCM REST-based API.

The functionality of the .NET library encapsulates the following:

- Present strongly-typed method signatures for supported functions
- Construct REST requests using the appropriate HTTP verbs and populate XML payload appropriately in the case of POST and PUT requests
- Inject the required headers for authentication by the DMCM server
- Inject the required header for the level of details requested by the client to the server

The library (all DLLs) is available at: https://github.com/dorongrinstein/dmcm-dotnet-library

The library was compiled to target .NET 3.5 (to accommodate the D1IM project)

The source code for the project is available at: https://github.com/dorongrinstein/enstratus-api-utils/tree/master/dotnet

Supported Methods

The public methods exposed by the library are self-explanatory:

- `public Client(string hostBase, string apiAccessId, string apiSecretKey, string userAgent, string apiRoot)`
- `public void AddHeader(string name, string value)`
- `public void clearHeaders()`
- `public CustomerList GetCustomerList()`
- `public string GetCustomersJson()`
- `public string GetAccountJson(string id)`
- `public string GetCloudsJson()`
- `public CloudList GetCloudList()`
- `public Account GetAccount(string id)`
- `public string CreateUser(string accountId,`

```
                                string givenName,
                                string familyName,
                                string email,
                                string emailTarget,
                                string notifyViaEmail,
                                string notifyViaScreen,
                                string eventType,
                                string severity,
                                string billingCode,
                                 string groupId)
```

- `public string CreateDeployment(DeploymentLaunch deployment)`
- `public string LaunchDeployment(string id)`
- `public string LaunchServer(string budget, string name, string description, string machineImageId, string product, string dataCenterId)`
- `public string StopServer(string serverId)`
- `public string TerminateServer(string serverId, string reason)`
- `public string GetAccountsJson()`
- `public AccountList GetAccountList()`
- `public string GetDeploymentsJson()`
- `public DeploymentList GetDeploymentList()`
- `public string GetServersJson()`
- `public ServerList GetServerList()`
- `public string invokeCommand(RestSharp.Method method, string resource, string parameters, object obj, RestSharp.Serializers.ISerializer serializer)`
- `public string invokeStringPost(string resource, string xml, Boolean put=false)`

Properties

The only property exposed by the Client class is:

- Details

Details is of type DetailsEnum which is defined as follows:

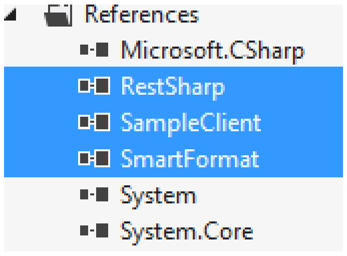
```
public enum DetailsEnum { none, basic, extended };
```

The default value of the Details property is basic. You can override the default by assigning another value supported by the DetailsEnum data type. Under the covers, the details property generates a DMCM **x-es-details** HTTP header property with a value of either “none”, “basic” or “extended”.

You can learn about this HTTP header as well as the entire DMCM REST API by reading this document: https://github.com/dorongrinstein/dmcm-dotnet-library/blob/master/REST%20API%20Specification-2013-03-13.pdf?raw=true

How to use the library in your own project

Step 1 – add reference to the 3 DLLs:



Step 2 – Call the constructor, passing in the string corresponding to your endpoint, API access ID, user agent (any value), API root and API secret key. Here are some valid values. Note these are just examples. You must obtain your own API key:

```
Host base: http://demo.enstratius.com:15000
API access Id: HUFVVXTGJWVZYFWRMAHU
User agent: test
API root: /api/enstratus/2013-03-13
```

Step 3 – You need to obtain the API ID and API Key from the DMCM web console.

Once the above 3 steps are completed you are ready to invoke the public methods of the client library. Simply add a using clause at the beginning of the code unit you’re writing (shown in C#):

```
using Dell.CTO.Enstratius;
```

Then, inside of a method, instantiate the Client component and call its methods. For example:

```
Client c = new Client("http://demo.enstratius.com:15000", "HUFVVXTGJWVZYFWRMAHU",
"/api/enstratus/2013-03-13", "test", "your api key goes here");

var servers = c.GetServerList();
```

Any questions

Contact Doron Grinstein

Email: doron.grinstein@software.dell.com

Mobile Phone: +1-818-822-5780

Skype: doron.grinstein1