Code Examples

See also this tutorial for more information about using the ASP.NET web API client libraries.

Making a GET request

Let's read orders created after a particular date. For security reasons, query results need to be paged, so we have to specify the $top option (and optionally $skip).

```
string publicKey = "0c6b33651708eb09c8a8d6036b79d739";
string secretKey = "3025c89ebaab20b71e0e42744239bf50";
string method = "get";
string accept = "application/json";
string timestamp = DateTime.UtcNow.ToString("o");        // 2013-11-11T10:15:54.1731069Z
string url = "http://localhost:1260/odata/v1/Orders?$top=10&$filter=CreatedOnUtc gt 2013-02-20T00:00:00Z";
```

First, we create the message representation.

```
var uri = new Uri(url);                // decode url
if (uri.Query != null && uri.Query.Length > 0)
{
        url = string.Concat(uri.GetLeftPart(UriPartial.Path), HttpUtility.UrlDecode(uri.Query));
}

var messageRepresentation = string.Join("\n",
           method.ToLower(),
           "",
           accept.ToLower(),
           url.ToLower(),
           timestamp,
         publicKey.ToLower()
);
```

It looks like:

```
get
application/json
http://localhost:1260/odata/v1/orders?$top=10&$filter=createdonutc gt 2013-02-20T00:00:00Z
2013-11-11T10:15:54.1731069Z
0c6b33651708eb09c8a8d6036b79d739
```

Now, we can calculate the HMAC signature by using our secret key.

```
 string signature = CreateSignature(secretKey, messageRepresentation);        //
hWce6V2KA0kkB0GBbIK0GSw5QAcS3+vj+m+WN/8k9EE=
```

We have all the information to set up the request, so we create a web request object and pass the required headers.

```
var request = (HttpWebRequest)WebRequest.Create(url);
request.Method = method;
request.UserAgent = "My shopping data consumer v.1.0";
request.Accept = accept;
request.Headers.Add("Accept-Charset", "UTF-8");
request.Headers.Add("SmartStore-Net-Api-PublicKey", publicKey);
request.Headers.Add("SmartStore-Net-Api-Date", timestamp);
request.Headers.Add("Authorization", "SmNetHmac1 " + signature);
```

The complete header looks like this:

```
User-Agent: My shopping data consumer v.1.0
Accept: application/json
Accept-Charset: UTF-8
SmartStore-Net-Api-PublicKey: 0c6b33651708eb09c8a8d6036b79d739
SmartStore-Net-Api-Date: 2013-11-11T10:15:54.1731069Z
Authorization: SmNetHmac1 hWce6V2KA0kkB0GBbIK0GSw5QAcS3+vj+m+WN/8k9EE=
```

Making a POST request

Posting means inserting data via API. This example shows how to add a new order note "Hello world!" to the order with ID 152. Here is the function to create the MD5 hash of the request body:

```
public string CreateContentMd5Hash(byte[] content)
{
          string result = "";
           if (content != null && content.Length > 0)
           {
                   using (var md5 = MD5.Create())
                   {
                           byte[] hash = md5.ComputeHash(content);
                           result = Convert.ToBase64String(hash);
                   }
           }
           return result;
}
```

No other variables have changed.

```
string content = "{\"OrderId\":152,\"Note\":\"Hello world!\",\"DisplayToCustomer\":false,\"CreatedOnUtc\":\"
2013-11-09T11:15:00\"}";

byte[] data = Encoding.UTF8.GetBytes(content);
string contentMd5Hash = CreateContentMd5Hash(data);

string method = "post";
string timestamp = DateTime.UtcNow.ToString("o");        // 2013-11-11T19:44:04.9378268Z
string url = "http://localhost:1260/odata/v1/OrderNotes";
```

We add the same header fields as in the previous example, and additionally:

```
request.ContentLength = data.Length;
request.ContentType = "application/json; charset=utf-8";
request.Headers.Add("Content-MD5", contentMd5Hash);        // optional
```

Then, we write the content into the request stream.

```
using (var stream = request.GetRequestStream())
{
          stream.Write(data, 0, data.Length);
}
```

The message representation is as follows:

```
post
lgifXydL3FhffpTIilkwOw==
application/json
http://localhost:1260/odata/v1/ordernotes
2013-11-11T19:44:04.9378268Z
0c6b33651708eb09c8a8d6036b79d739
```

The header looks like this:

```
User-Agent: My shopping data consumer v.1.0
Accept: application/json
Accept-Charset: UTF-8
SmartStore-Net-Api-PublicKey: 0c6b33651708eb09c8a8d6036b79d739
SmartStore-Net-Api-Date: 2013-11-11T19:44:04.9378268Z
Content-Type: application/json; charset=utf-8
Content-Length: 100
Content-MD5: lgifXydL3FhffpTIilkwOw==
Authorization: SmNetHmac1 ejKxxtHNJYHCtBglZPg+cbSs3YTrA50pkfTHtVb1PMo=
```

As a general rule, POST, PUT and PATCH are returning the added or changed record. For example:

```
{
    "odata.metadata":"http://localhost:1260/odata/v1/$metadata#OrderNotes/@Element",
    "OrderId":152,
    "Note":"Hello world!",
    "DisplayToCustomer":false,
    "CreatedOnUtc":"2013-11-09T11:15:00","Id":692
}
```

Processing the response

Example of reading the response into a string:

```
HttpWebResponse webResponse = null;
string response;

try
{
         webResponse = request.GetResponse() as HttpWebResponse;
    using (var reader = new StreamReader(webResponse.GetResponseStream(), Encoding.UTF8))
    {
             response = reader.ReadToEnd();
    }
}
catch (WebException wexc) { /* ... */ }
catch (Exception exc) { /* ... */ }
finally
{
          if (webResponse != null)
           {
                   webResponse.Close();
                   webResponse.Dispose();
           }
}
```

JSON data can be easily parsed into dynamic or strongly typed objects using Json.NET. This example deserializes a JSON string into a list of customers.

```
 public class Customer
{
           public string Id { get; set; }
           public string CustomerGuid { get; set; }
           public string Email { get; set; }
        // more properties...
}

JObject json = JObject.Parse(response);
string metadata = (string)json["odata.metadata"];

if (!string.IsNullOrWhiteSpace(metadata) && metadata.EndsWith("#Customers"))
{
           List<Customer> customers = json["value"].Select(x => x.ToObject<Customer>()).ToList();
}
```

Dynamic JSON parsing might look like this:

```
dynamic dynamicJson = JObject.Parse(response);

foreach (dynamic customer in dynamicJson.value)
{
        string str = string.Format("{0} {1} {2}", customer.Id, customer.CustomerGuid, customer.Email);
        Debug.WriteLine(str);
}
```