Search SOAP API - CES 7.0

Search SOAP API Home

The Coveo Platform 7.0 comes with a Coveo Search SOAP API that you can use to perform searches from custom search interfaces using Java, Python, .Net, or C++. This web service is accessible from the Coveo Master server by default on port 52810.

Note:
You can specify a different port when you create or import the index (see Creating a New Index or Importing an Existing CES 6 Index).

You can get the descriptions of the functionality offered by the Coveo Search Web Service in a Web Service Description Language (WSDL) form using the following URL:

`https://CoveoMasterServer:52810/7.0/CoveoSearchService?wsdl`

You can review the Search SOAP API reference topics (see Search SOAP API - Methods Schema).

Contact the Coveo Professional Services for more information or assistance in using the Coveo Search SOAP API.

Getting Started Using the Coveo Search SOAP API

This topic describes the steps needed to perform queries through the Coveo Search Web Service (WSDL) using Microsoft Visual Studio 2010.

The complete code used in this tutorial:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using WebServiceSample.CoveoSearchServiceAPI;
using System.Security.Cryptography.X509Certificates;
using System.ServiceModel;
using System.Net;
namespace WebServiceSample
{
    class Program
    {
        static void Main(string[] args)
        {
            // Create a service reference using
https://yourservername:52810/7.0/CoveoSearchService?wsdl
            var service = new CoveoSearchServiceClient();
            // Once you've created an index, you can use the default certificate.
            var sslCertificatePath = @"C:\CES7\Config\Certificates\cert-iis.p12";
            var certificate = new
System.Security.Cryptography.X509Certificates.X509Certificate2(sslCertificatePath,
"CoveoCert", X509KeyStorageFlags.MachineKeySet);

            // Setup the certificate on the endpoint.
            BasicHttpBinding binding = (BasicHttpBinding) service.Endpoint.Binding;
            binding.Security.Mode = BasicHttpSecurityMode.Transport;
            binding.Security.Transport.ClientCredentialType =
HttpClientCredentialType.Certificate;
            service.ClientCredentials.ClientCertificate.Certificate = certificate;
            ServicePointManager.ServerCertificateValidationCallback += (p_Sender,
p_Certificate, p_Chain, p_SslPolicyErrors) => true;

            // Create a UserId.
            var userId = new UserId();
            userId.Provider = "Active Directory";
            userId.Name = ; // Insert the Active Directory user name with the domain
name (domain\username) that should execute the search query (the logged in user). It
should be a valid Active Directory user.
```

```
            userId.Type = UserIdType.User;
            //userId.Type = UserIdType.Group; // If the user is a group.

            // Create a search session with the UserId created before
            var searchSession = new SearchSession();
            searchSession.UserId = userId;

   // Create a query.
            var query = new QueryParams();
            query.NumberOfResults = 10;
            query.Expression = "@uri";

   // Add a facet to the query if needed.
   var facet = new GroupByField();
   facet.FieldName = "@sysfiletype";
   facet.MaxNumberOfGroupByResults = 5;
   facet.SortCriteria = GroupBySortCriteria.SortByOccurrence;
   query.GroupByFields = new GroupByFieldV();
   query.GroupByFields.Add(facet);

   // Execute the query.
            var res = service.ExecuteQuery(searchSession, query);

           // Then you can iterate on the res.Results or res.GroupByResults to do what
you need to do!
        }
    }
```

```
}
```

Adding Coveo Web Service Reference

You first need to create/open a project and add the service reference.

1. Create/Open your Visual Studio project.
2. In the **Solution Explorer**, right click **Service References**.
3. Click **Add Service Reference**.
4. In the **Address** field, enter the URL of the web service, and then click **Go**. A **Security Alert** dialog may appear, click **Yes** to proceed.

   The web service URL should look like:

   ```
   https://yourservername:52810/7.0/CoveoSearchService?wsdl
   ```

   `CoveoSearchServiceAPI` should appear under **Services**.
5. In **Namespace**, enter a meaningful name (ex.: `CoveoSearchServiceAPI`), and then click **OK** to add the reference.

Dependencies

To use the Coveo Web Service, you need to include at least these references in your code:

```
// Basic C# references or other references...
...

// Coveo Web Service (WebServiceSample.TheNameYouGaveIt)
using WebServiceSample.CoveoSearchServiceAPI

// Certificate
using System.Security.Cryptography.X509Certificates;

// Setup the certificate on the endpoint
using System.ServiceModel;
using System.Net; // ServicePointManager class

namespace WebServiceSample
{
    class Program
    {
        static void Main(string[] args)
        {
   ...
        }
    }
}
```

Initializing the Server Client

Now that you have included everything you can start creating and using your service reference.

You initialize a service by calling the `CoveoSearchServiceClient()` method and retrieving what it returns into a variable:

```
// Create a service reference using
https://yourservername:52810/7.0/CoveoSearchService?wsdl
var service = new CoveoSearchServiceClient();
```

Using the Default Certificate

The next step is to create a certificate.

```
// Once you've created an index, you can use the default certificate.
var sslCertificatePath = @"C:\CES7\Config\Certificates\cert-iis.p12";
var certificate = new
System.Security.Cryptography.X509Certificates.X509Certificate2(sslCertificatePath,
"CoveoCert", X509KeyStorageFlags.MachineKeySet);
```

The `sslCertificatePath` written above is the default path. The real path depends on where you installed CES, it should be:

`[Index_Path]\Config\Certificates\cert-iis.p12`

Creating a Session With the Current User

To perform a query, you need to create a session with a user.

You can create a user by providing a security provider, a name, and a type to a new user.

```
// Create a UserId.
var userId = new UserId();
userId.Provider = "Active Directory";
userId.Name = ; // Insert the Active Directory user name with the domain name
(domain\username) that should execute the search query (the logged in user). It should
be a valid Active Directory user.
userId.Type = UserIdType.User;
//userId.Type = UserIdType.Group; // If the user is a group.

// Create a search session with the UserId created before
var searchSession = new SearchSession();
searchSession.UserId = userId;
```

Creating a Query

To execute a query, you first need to initialize a query variable. You can then specify the number of results that you want, the expression of the query (in this case `@uri` will return all results), etc.

You can find the list of query parameters in the Coveo online help (see QueryParams Complex Type).

```
// Create a query
var query = new QueryParams();
query.NumberOfResults = 10;
query.Expression = "@uri";
//query.Expression = "@txtanpopular=" + (tag ?? "popularusa") + "
@templatename=CarModel"; // An example of a more complex query expression.
```

Adding a Facet to the Query

Before executing your query, you might want to add facets to your query. Here is how you can do it:

```
// Add a facet to the query
var facet = new GroupByField();
facet.FieldName = "@sysfiletype";
facet.MaxNumberOfGroupByResults = 5;
facet.SortCriteria = GroupBySortCriteria.SortByOccurrence;
query.GroupByFields = new GroupByFieldV();
query.GroupByFields.Add(facet);
```

Execute the Query

You can finally execute the query by calling the `ExecuteQuery` method.

```
// Execute the query
var res = service.ExecuteQuery(searchSession, query);
```

The results of a query can be stored in a variable. In the above example, it is stored in the variable named `res`.

You can then access any information about the result of the query from this variable.

Configuration

To ensure proper functioning of the web service, you may have to change some properties in your `app.config` file, or if you are creating a web application, in your `web.config` file.

When you add your Service Reference, Visual Studio automatically creates for you a `.config` file.

Consider changing the following two properties to ensure that your queries work properly:

- `maxBufferSize`
- `maxReceivedMessageSize`
  Ensure that both properties are set high enough to prevent your queries to return errors.
  Refer to the following MSDN documents for more information:
    - BasicHttpBinding.MaxBufferSize Property
    - BasicHttpBinding.MaxReceivedMessageSize Property

To change the values of these properties

In the .config (app.config or web.config) file you need to add maxBufferSize and maxReceivedMessageSize to CoveoSearchServiceBinding. The default value is 65536. Your file should look something like this afterwards (with real values instead of `ENTER_A_VALUE`):

```
...
        <bindings>
            <basicHttpBinding>
                <binding name="CoveoSearchServiceBinding"
maxBufferSize="ENTER_A_VALUE" maxReceivedMessageSize="ENTER_A_VALUE">
                    <security mode="Transport" />
                </binding>
                <binding name="CoveoSearchServiceBinding1" />
            </basicHttpBinding>
        </bindings>
...
```