# Virtual Agent Integration with Alchemy API

## Note about references

The following examples use the following references. These examples also use a reference to the Alchemy sdk.dll.

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Xml.Linq;

using System.Xml.Serialization;

## Connect to the Alchemy service

The following code shows you how to authenticate with the Alchemy AI service. The only credential this requires is your Alchemy API key which can be requested from <http://www.alchemyapi.com/api/register.html>.

AlchemyAPI alchemyAPI = new AlchemyAPI.AlchemyAPI();

public void ConnectToAlchemy()

{

try

{

alchemyAPI.SetAPIKey(apiKey);

isConnectedToAlchemy = true;

}

catch (Exception ex)

{

Console.WriteLine(ex.ToString());

}

}

## Get ranked keywords from some text

One of the things Alchemy API is very good at is analyzing a body of text. This first example will show you how to leverage this capability to extract topic keywords from some text, ranked by their relevance as perceived by the Alchemy service. Note, Alchemy responses are in XML format. For brevity, the RankedKeywords class that is used to bind to the deserialized response is not included, but you can construct your own class from the resources provided by Alchemy on this page (scroll to the bottom Response Format (XML) section) <http://www.alchemyapi.com/api/keyword/textc.html>. I’ve also excluded the TopFiveKeywords method, this simply string-ifies the text field of the first five keywords.

public RankedKeywords.Response GetRankedKeywords(String text)

{

RankedKeywords.Response status = new RankedKeywords.Response();

try

{

if (alchemyAPI != null)

{

RankedKeywords.results response = new RankedKeywords.results();

XmlSerializer serializer = new XmlSerializer(response.GetType());

var xml = alchemyAPI.TextGetRankedKeywords(text);

Console.WriteLine(xml);

object deserialized = serializer.Deserialize(ToStream(alchemyAPI.TextGetRankedKeywords(conversationText)));

response = (RankedKeywords.results)deserialized;

status.Result = response;

status.Status = "SUCCESS";

Console.WriteLine("Ranked Keyword Count: " + status.Result.keywords.Count());

Console.WriteLine("Top 5 Keywords: " + TopFiveKeyWords(status.Result.keywords));

}

}

catch (Exception ex)

{

status.Result = null;

status.Status = "ERROR: " + ex.ToString();

Console.WriteLine(ex.ToString());

}

return status;

}

## Get sentiment analysis for some text

This second example will show you how to use Alchemy API to retrieve a sentiment analysis of a body of text. Alchemy uses the following data points to report sentiment analysis:

* Type – sentiment polarity: "positive", "negative", or "neutral"
* Score - sentiment strength (0.0 is neutral)
* Mixed - whether sentiment is mixed (both positive and negative) (1 is mixed)

Once again, for brevity we have excluded the class source code that binds to the Alchemy XML response. This documentation shows you how the response will be formed <http://www.alchemyapi.com/api/sentiment/textc.html>.

public SentimentAnalysis.Response GetSentiment(string conversationText)

{

SentimentAnalysis.Response status = new SentimentAnalysis.Response();

try

{

if (alchemyAPI != null)

{

SentimentAnalysis.results response = new SentimentAnalysis.results();

XmlSerializer serializer = new XmlSerializer(response.GetType());

object deserialized = serializer.Deserialize(ToStream(alchemyAPI.TextGetTextSentiment(conversationText)));

response = (SentimentAnalysis.results)deserialized;

status.Result = response;

status.Status = "SUCCESS";

if (status.Result != null && status.Result.docSentiment != null)

{

if (status.Result.docSentiment.type != null)

{

Console.WriteLine("Sentiment type: " + status.Result.docSentiment.type);

}

Console.WriteLine("Sentiment score: " + status.Result.docSentiment.score);

var mixed = status.Result.docSentiment.mixed;

Console.WriteLine("Mixed sentiment: " + (mixed == 1 ? "True" : "False"));

}

}

}

catch (Exception ex)

{

Console.WriteLine(ex.ToString());

}

return status;

}