using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Net;

using System.IO;

using System.Windows.Forms;

namespace Eyespy\_Core\_v1

{

/// <summary>

/// Used to interact with Eyespy HQ, through this class are submitted URL's and metadata to the HQ datacenter.

/// </summary>

/// <remarks></remarks>

class HTTPWriter

{

/// <summary>

/// Once the URL has passed all the validation stages, it can be posted to the HQ via this procedure.

/// </summary>

/// <param name="URL">The URL to add to the database.</param>

/// <param name="meta">The keywords realting to the URL to add to the database.</param>

/// <remarks></remarks>

public void submitToHQ(string URL, string meta)

{

HttpWebRequest request = (HttpWebRequest)WebRequest.Create("http://www.charlesnet.info/eyespy/submit.php"); // create a virtual web request object

string data = "URL=" + URL + "&keywords=" + meta; // this is standard HTML-parsed data in the form: VARIABLE="DATA" & VARIABLE2="DATA2", extractable via a GET\_ request by the target PHP script

byte[] postData = Encoding.UTF8.GetBytes(data); // all data must be sent as a byte-array, using the embedded C# Encoding class to convert the data

// a static usage is more appropriate in this case as it's the only usage made of this class

request.Method = "POST"; // variable of the HttpWebRequest class, the Method in which we wish to post our application.

request.ContentType = "application/x-www-form-urlencoded"; // standard HTML-header cotent type

request.ContentLength = postData.Length; // size of the body of data (in this case, the size of both arguments inside the data string defined above

using (var stream = request.GetRequestStream()) // use a stream writer to send a stream of bytes to the URL.

{

stream.Write(postData, 0, postData.Length); // send the converted data to the source URL

}

var responce = (HttpWebResponse)request.GetResponse(); // this is the opposite Class of the HtmlWebRequest Class, the assignment needs to be type-casted

// as we wish to get the Responce from the Request, readable only via the HttpWebResponce class.

var responceString = new StreamReader(responce.GetResponseStream()).ReadToEnd(); // get the string 'answer' from the Request, again using a stream reader to convert the

// inbound bytes to a string.

// the purpose of the responceString is purely for debugging, in the context of the final product, the above line of code can be ignored.

}

}

}