BAI3110 – Web Services Assignment _____

Name

Referring to: http://www.w3.org/TR/soap/

http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

1. What is the main purpose of SOAP?

_____

2. What is the current version of SOAP?

_____

3. What is the difference between a Simple Type and Complex Type?

_____

_____

4. List 3 simple types.

_____

_____

_____

5. What MUST a Soap client do when requesting SOAP over HTTP?

_____

_____

6. Can you transfer Binary information with Soap? If so, how?

_____

_____

7. Demonstrate a small test client calling a webservice from http://www.xmethods.org.

Seeing that Xmethods.org is pretty much dead this morning, use the direct URL: http://ws.cdyne.com/IP2Geo/IP2Geo.asmx?WSDL

- ~~Select Full List at the bottom of the page in your browser.~~
- ~~Type Control-F to search and type in "IP2Geo". Note the Description and then click the IP2Geo hyper link. Click the WSDL Hyperlink.~~
- Save the WSDL URL in Notepad for later.
- Edit the URL in the browser and delete the "?wsdl" characters and hit enter. This will bring up a description page. Note that there is only one method. Click it. This brings up a Test page for the method.
- Enter an IP address. Enter an IP address of 50.63.202.1 (www.elyk.net), and a license key of "0". Note the information returned.
- Create an empty WebSite named BAIS3110WebServiceTest, and add a WebForm to the solution.
- WebSite -> Add Service Reference -> Advanced (Button) -> Add Web Reference. Enter the WSDL URL you saved in Bullet 3. Click the green arrow then click the "*Add Reference"* Button. This will create a Proxy object that will manage the WebService for us.

  Note the Web Reference Name, located above the Add reference button and make sure it is **com.cdyne.ws**

- In the Page_Load method add the following code:

```
com.cdyne.ws.IP2Geo local = new com.cdyne.ws.IP2Geo();
com.cdyne.ws.IPInformation anAddress;

anAddress = local.ResolveIP("50.63.202.1 ", "0");

Response.Write(anAddress.City.ToString() + "<br />");
Response.Write(anAddress.Country.ToString() + "<br />");
Response.Write(anAddress.Latitude.ToString() + "<br />");
Response.Write(anAddress.Longitude.ToString() + "<br />");
```

- Note the Output
- If you would like more practice in calling webservices try calling another service such as: http://wsf.cdyne.com/WeatherWS/Weather.asmx and work out the necessary code on your own.

8. Use Fiddler to capture an HTTP packet from Step 7, and record relevant information on the back of this page.

Bonus

9. Create a small C# console app that connects to Microsoft's Bing Translation service as demonstrated here: http://code.msdn.microsoft.com/Walkthrough-Translator-in-7e0be0f7

- Sign up as an Azure Developer; pick the free plan, no payment required!
- Create a C# Console Application, as directed; I suggest a project name "RandomTranslator".
- Download the Proxy as directed, right click "References" in your Solution Explorer and select "Add Reference". Select System.Data.Services.Client and click ok.
- Select the Browse Code Tab in the Walkthrough page, click RandomTranslator and then Program.cs to get the code to place inside of Main() and the methods after Main(). Don't just copy everything as it will mangle your namespace and class declaration if you did not select "RandomTranslator" in Bullet 2.
- Replace the accountKey in InitializeTranslatorContainer() with your own AND comment the Throw new Exception() immediately following.
- Project Menu -> Project Properties -> Debug, and enter a test string in the Command Line Arguments.