SER422 Spring 2018 ONLINE Lab 4 – SOA **Due: Monday, April 16 2018 at 11:59:00pm**

Task 1: (50 points) Construct a Web Services Client for an existing Web Service

The WSDL at http://vhost3.cs.rit.edu/weather/Service.svc?wsdl defines a 3-method service for weather. Your task is to write a servlet-based web application client for this service. Your servlet-based client should provide an interface for obtaining the various parameters needed to make the client calls. You can see a model of the client at http://vhost3.cs.rit.edu/Application/ click on the "Weather" tab.

Some very important things to note:

1. The website with the Weather service has documentation and C# code for the service, and I am perfectly fine with you looking at it. However the code and documentation is for implementing the service, so it doesn't really help you much – you are constructing a client.
2. I suggest constructing a main program first before worrying about servlet integration. I was able to get a client going in literally 5 minutes using Eclipse's Dynamic Web Project wizard.
3. You may use either Eclipse's Dynamic Web Project wizard or axis2's wsdl2java tools for this.
4. You are also free to screen-scrape the client on the RIT site – it has a simple web form you can grab and embed as an HTML form in your client web application. I do not require the Map that they show (but see the extra credit below). However, you are not required to use their UI either, but you need to show at least as many of the response attributes as they do.
5. If you look, you see that the SOAP responses actually wrap a RESTful response formatted in JSON, so you will need to parse that JSON to get returned weather data.

Task 2: (50 points) Create your own Web Service

You are given an example web application "lab4given". This is a simple servlet that uses Lab4Service to get numeric grades and mapping to letter grades. The servlet acts as a driver program to test the service. The Lab4Service calculates numeric scores for a mock set of Students who are in various majors ("History", "Engineering", "English", "Nursing", or "Psychology") and various years of study ("Freshman", "Sophomore", "Junior", "Senior", or "Graduate"). The calculateGrade method takes a String indicating a year in school, OR a String subject, OR both and returns a double value that is the average score of that cohort.

You are asked to make Lab4Service a web service.

Your requirements:

1. Add a method named getSubjects() to the service that returns an array of Strings with the distinct subjects available.
2. Use the java2wsdl tool to generate WSDL for Lab4Service.
3. Use the wsdl2java tool to generate the skeleton code for the service.
4. Refactor the generated skeleton code to use Lab4Service to implement the web service.
5. Host the service in your axis2.jar

Some very important things to note:

1. You will need to submit your generated .aar file, which should be renamed to lab4-<asurite>.aar. Axis2 tools will auto-generate a build.xml for you but you will need to make sure the service gets named appropriately so we don't have name conflicts when we grade.

Extra Credit:

1. (20 points) In your client in Task 1 I said you don't have to implement the map feature that the RIT webpage shows. That map feature shows a location but no weather data. The actual weather data service being used is Weather Underground, and on their API page they provide an ability to get images as radar, satellite, or both. Augment your Task 1 solution to retrieve either or both of these based on query string parameters. Note you do not have to do this as a SOAP-based web service, but rather acting as a direct client. Basically you are creating a server-side mashup. Note you will need an APIkey, but you can get an "Anvil Plan" API key for free which includes radar, satellite thumbnails, and animated versions of both (the animated versions would be very cool!).
2. (10 points) The current service code for Lab4Service in Task 2 does not handle errors very well. If you give an invalid String for year or major you will degenerate to a failed if test in Lab4Service.java and return 0.0 and "E" for calculateGrade and mapToLetterGrade respectively. Modify the code so that bad parameters are detected and an error returned, and have your service return a fault in the SOAP message.

SUBMISSION:

- Your submission should be named lab4.<asurite>.zip for upload to Blackboard.
- If you do an EC, submit a separate zipfile named lab4_ec_task#.asurite.zip. I prefer this just to make sure your EC version does not interfere with a properly working Task 1 or 2 solution.
- You may submit as many times as you like, there is NO reason to to ask for a late submission!
- If you use jar to archive your files, please be sure to change the extension from .jar to .zip before submitting.
- Please do not bundle in your Axis2 libraries, we have them and we do not want your submission to be huge!
- Please put a README.txt in the root of your project directories.