Building a WebService Test Plan

In this lesson, you will learn how to create a basic Test Plan to test a Web Service. You will create five users that send requests to one page. Also, you will tell the users to run their tests twice. So, the total number of requests is (5 users) x (1 requests) x (repeat 2 times) = 10 HTTP requests.

1. Adding Users

The first step you want to do with every JMeter Test Plan is to add a Thread Group element. The Thread Group tells JMeter the number of users you want to simulate, how often the users should send requests, and the how many requests they should send.

Go ahead and add the ThreadGroup element by first selecting the Test Plan, clicking your right mouse button to get the Add menu, and then select Add → ThreadGroup.

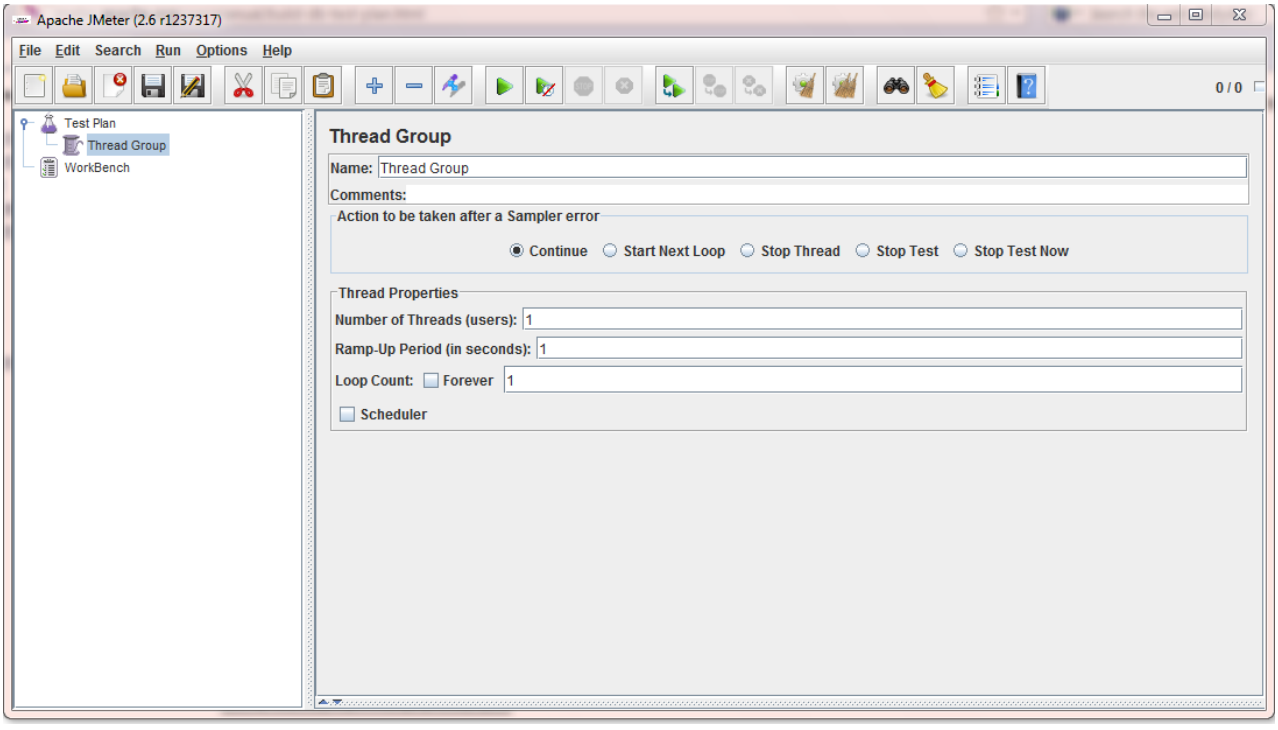

Figure 1.1. Thread Group with Default Values

Providing a more descriptive name for the Thread Group as 'JMeter Users'.

Next, increase the number of users to 5.

In the next field, the Ramp-Up Period, leave the default value of 0 seconds. This property tells JMeter how long to delay between starting each user. For example, if you enter a Ramp-Up Period of 5 seconds, JMeter will finish starting all of your users by the end of the 5 seconds. So, if we have 5 users and a 5 second Ramp-Up Period, then the delay between starting users would be 1 second (5 users / 5 seconds = 1 user per second). If you set the value to 0, then JMeter will immediately start all of your users.

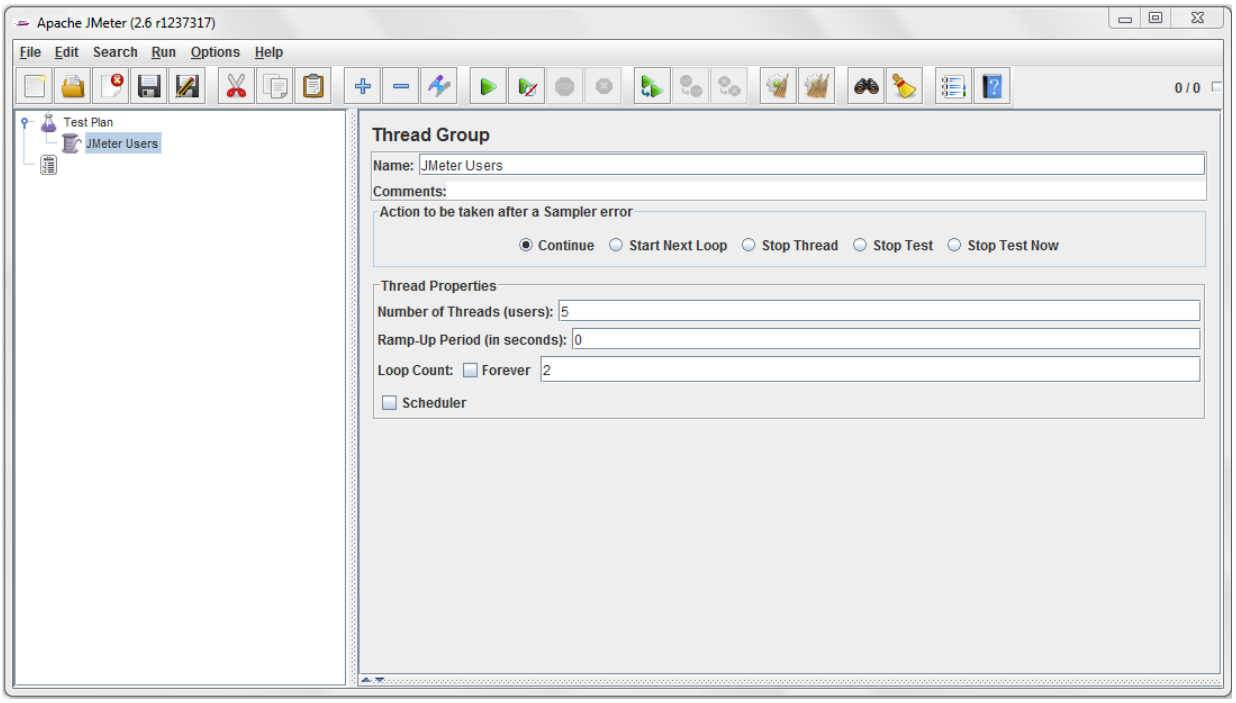

Figure 1.3 JMeter Users Thread Group

2. Adding Web Service Requests

In our Test Plan, we will use a .NET web service. Since you're using the web service sampler, we won't go into the details of writing a web service. If you don't know how to write a web service, Google for web service and familiarize you with writing web services for Java and .NET. It should be noted there is a significant difference between how .NET and Java implement web services. The topic is too broad to cover in the user manual. Please refer to other sources to get a better idea of the differences.

Start by adding the sampler Web Service (SOAP) Request to the Jakarta Users element (Add → Sampler → Web Service (SOAP) Request). Then, select the web service Request element in the tree and edit the following properties.

1. Change the Name field to "Web Service (SOAP) Request".
2. Enter the WSDL URL and click "Load WSDL".

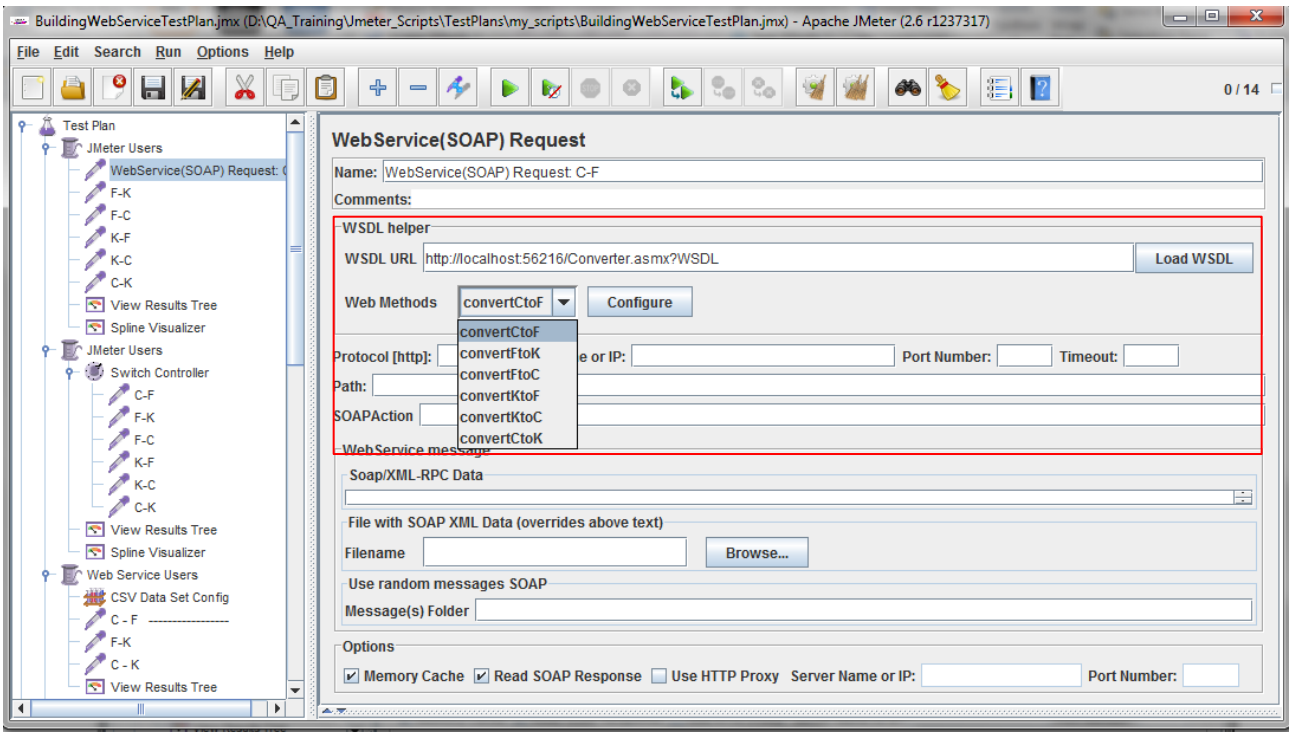

Figure 1.4. Web Methods

If the WSDL file was loaded correctly, the "Web Methods" drop down should be populated. If the drop down remains blank, it means there was a problem getting the WSDL. You can test the WSDL using a browser that reads XML. For example, if you're testing an IIS web service the URL will look like this: http://localhost/myWebService/Service.asmx?WSDL. At this point, SOAP Action, URL and SOAP Data should be blank.

Next, select the web method and click "Configure". The sampler should populate the "URL" and "SOAP Action" text fields. Assuming the WSDL is valid, the correct soap action should be entered.

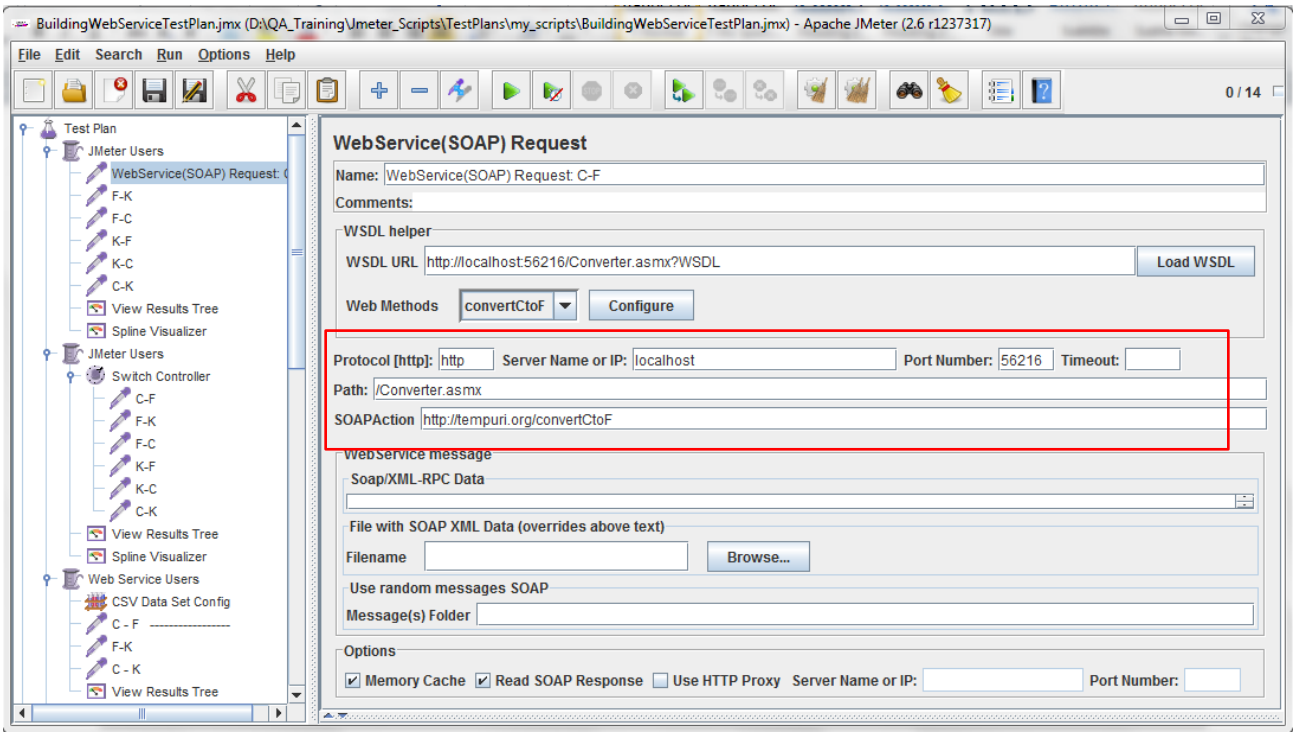

Figure 1.5 Configure Web Method

The last step is to paste the SOAP message in the "SOAP/XML-RPC Data" text area. You can optionally save the soap message to a file and browse to the location. For convenience, there is a third option of using a message folder. The sampler will randomly select files from a given folder and use the text for the soap message.

If you do not want JMeter to read the response from the SOAP Web service, uncheck "Read Soap Responses." If the test plan is intended to stress test a web service, the box should be unchecked. If the test plan is a functional test, the box should be checked. When "Read Soap Responses" is unchecked, no result will be displayed in view result tree or view results in table.

An important note on the sampler. It will automatically use the proxy host and port passed to JMeter from command line, if those fields in the sampler are left blank. If a sampler has values in the proxy host and port text field, it will use the ones provided by the user. If no host or ports are provided and JMeter wasn't started with command line options, the sampler will fail silently. This behavior may not be what users expect.

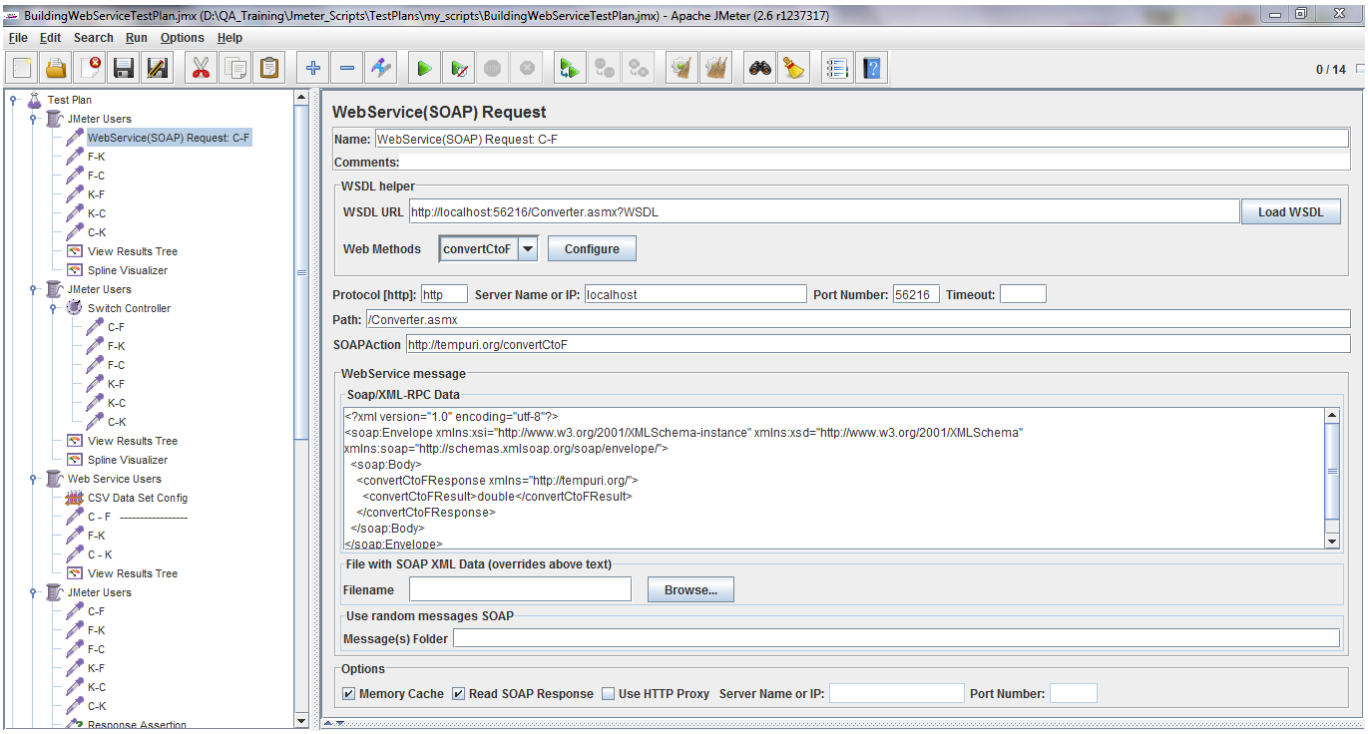

Figure 1.8. Web Service Message

3. Adding a Listener to View/Store the test Results

The final element you need to add to your Test Plan is a Listener. This element is responsible for storing all of the results of your HTTP requests in a file and presenting a visual model of the data.

Select the JMeter Users element and add a View Results Tree (Add → Listener → View Results Tree).

Before execute the JMeter test plan you need to run the web service.

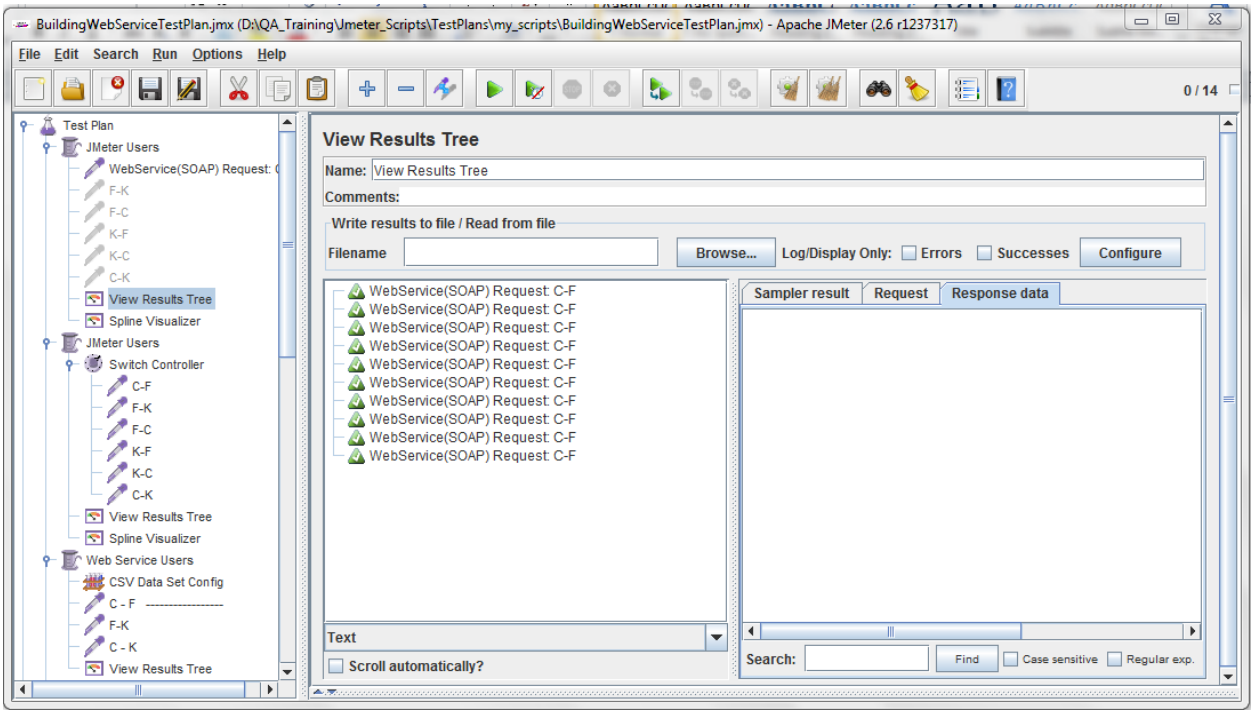

Figure 1.9. View Results Tree