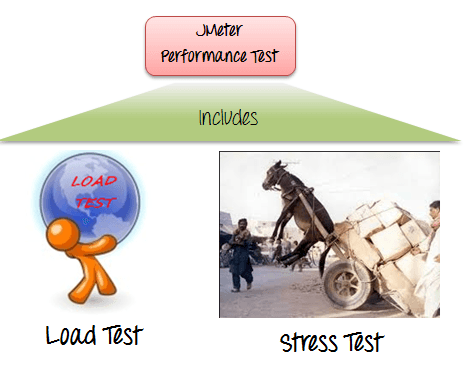
**Performance Testing using Jmeter**

Performance Testing is crucial to determine that the web application under test will satisfy **high load** requirements. It can be used to analyze overall server performance under heavy load.

JMeter testing tool offers following **benefit** in Performance Testing

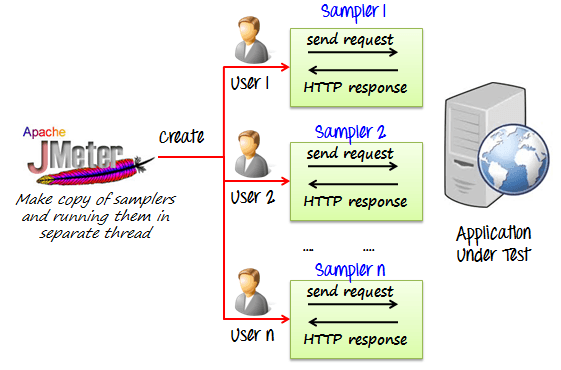
* JMeter can be  used to test performance of both **static** resources such as JavaScript and HTML, as well as **dynamic** resources, such as JSP, Servlets, and AJAX.
* JMeter can **discover** maximum number of concurrent users that your website can handle
* JMeter provides a variety of graphical analyses of performance reports.

JMeter Performance Testing includes:



* **Load** Testing: Modeling the expected usage by simulating multiple user access the Web services concurrently.
* **Stress** Testing: Every web server has a maximum load capacity. When the load goes beyond the limit, the web server start responding slowly and produce errors. The purpose of the Stress Testing is to find the maximum load the web server can handle.

The figure below shows how JMeter simulates the heavy load :



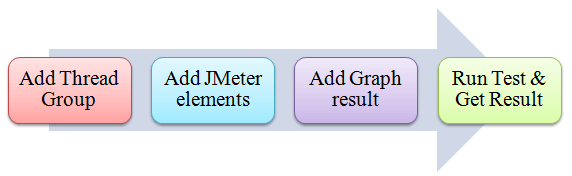
**Create a Performance Test Plan in Jmeter**

In this tutorial, we are doing a performance analysis of Google.com for 1000 users

Before testing the performance of target web application, we should determine-

* **Normal Load**: Average number of users visit your website
* **Heavy Load**: The maximum number of users visit your website
* What is your **target** in this test?

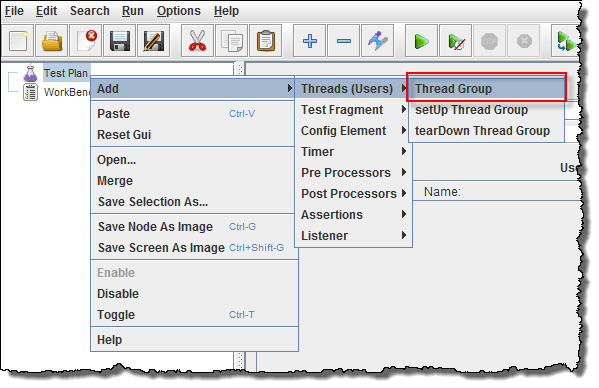
Here is the **roadmap** of this practical example



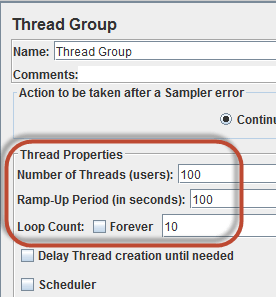
**Step 1) Add Thread Group**

1. Start **JMeter**
2. Select **Test Plan** on the tree
3. Add **Thread Group**

Right click on the Test Plan and add a new thread group: **Add**-> **Threads (Users)** -> **Thread Group**

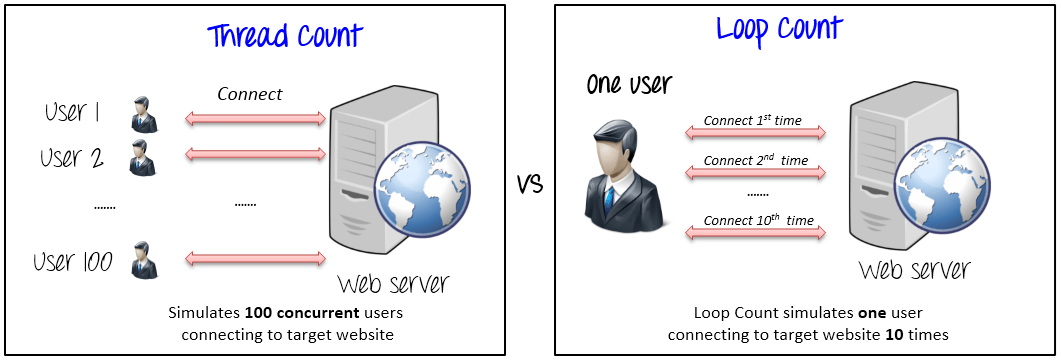


In Thread Group control panel, enter Thread Properties as following:

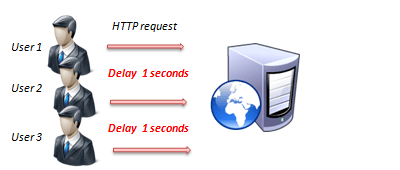


* **Number of Threads**: 100 (Number of users connects to target website: 100)
* **Loop Count**: 10 (Number of time to execute testing)
* **Ramp-Up Period**: 100

The Thread Count and The Loop Counts are **different.**



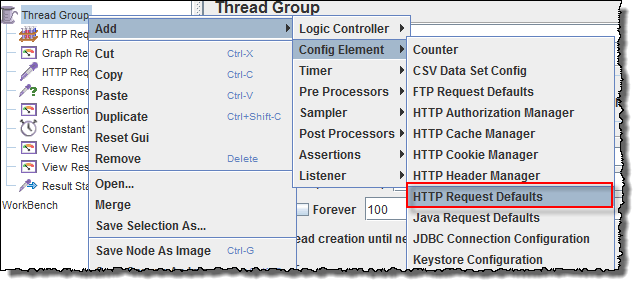
Ramp-Up Period tells JMeter how long to **delay** before starting next user. For example, if we have 100 users and a 100 second Ramp-Up period, then the delay between starting users would be 1 second (100 users /100 seconds)

****

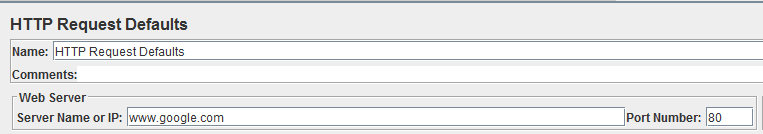
**Step 2) Adding JMeter elements**

### HTTP request Default

This element can be added by right-clicking on the Thread Group and selecting: **Add**->**Config Element**->**HTTP Request Defaults.**

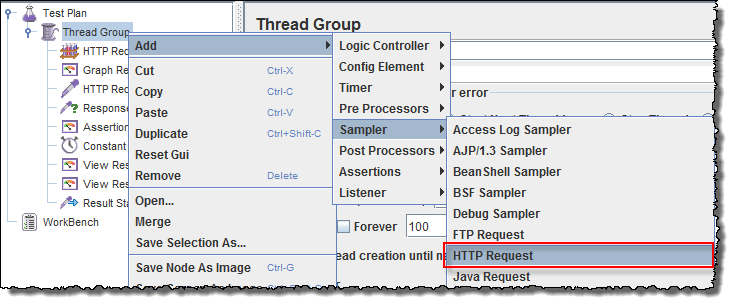


In the HTTP Request Defaults control panel, enter the Website name under test (http://www.google.com)

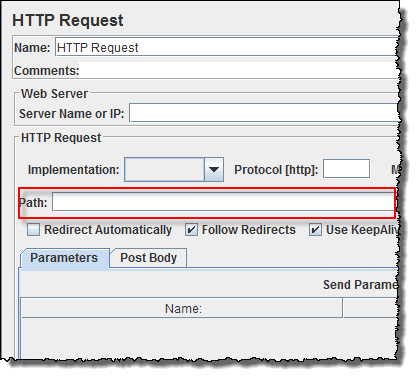


### HTTP Request

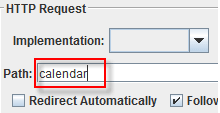
Right-click on Thread Group and select: **Add**->**Sampler**->**HTTP Request**.



In HTTP Request Control Panel, the Path field indicates which **URL request** you want to send to Google server.



For example, if you enter "calendar" in Path field. JMeter will create the URL request http://www.google.com/calendar  to Google server



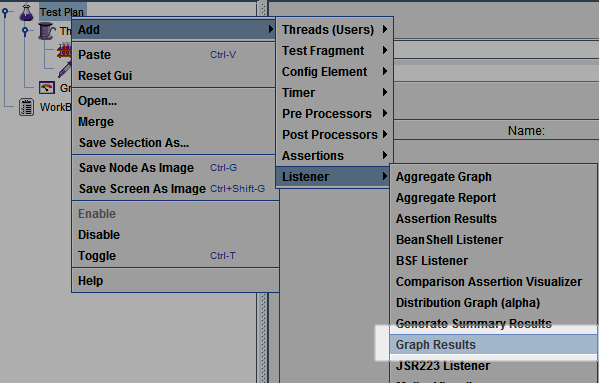
If you keep  the Path field blank  JMeter will create the URL request http://www.google.com to Google server.

**In this test, you keep the Path field blank to make JMeter create the URL request http://www.google.com to Google server.**

## Step 3) Adding Graph result

JMeter can show the test result in Graph format.

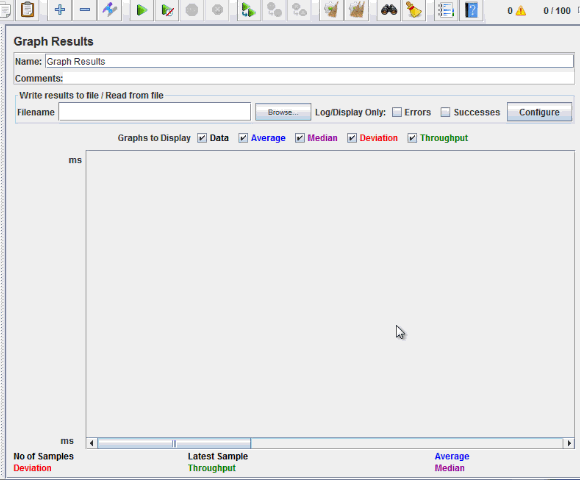
Right click Test Plan, **Add**->**Listener**->**Graph Results**



## Step 4) Run Test and get the test result

Press **Run** button (Ctrl + R) on Toolbar to start the software testing process. You will see the test result display on Graph at the real time.

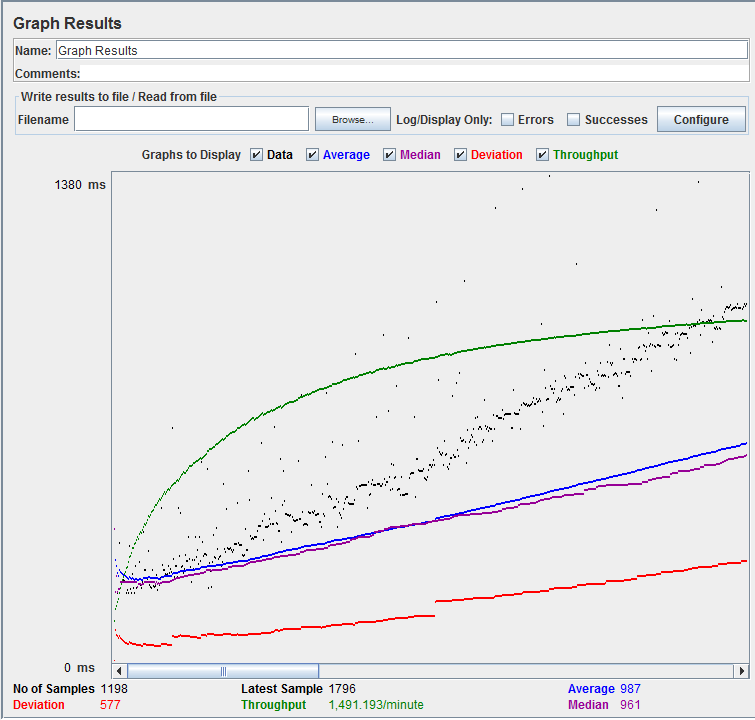
The picture below presents a graph of a test plan, where we simulated 100 users who accessed on website www.google.com.



At the bottom of the picture, there are the following statistics, represented in colors:

* Black: The total number of current samples sent.
* Blue: The current average of all samples sent.
* Red: The current standard deviation.
* Green: Throughput rate that represents the number of requests per minute the server handled

Let analyze the performance of Google server in below figure.



To analyze the performance of the web server under test, you should focus on 2 parameters

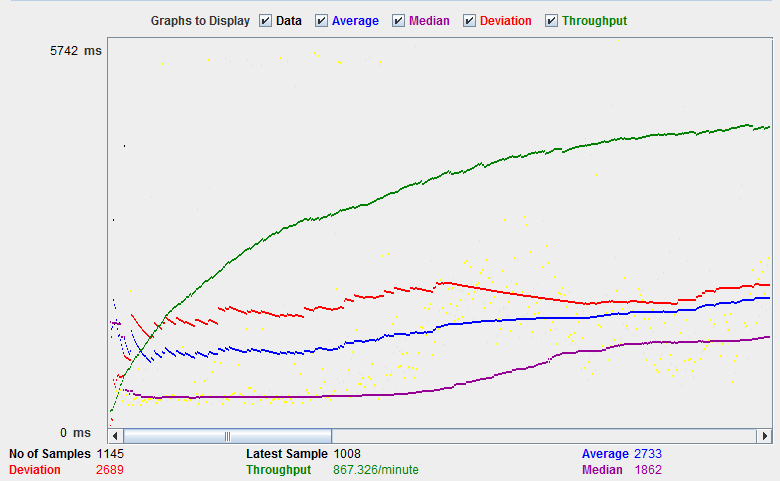
* Throughput
* Deviation

The **Throughput** is the most important parameter. It represents the ability of the server to handle heavy load.  The **higher** the Throughput is, the **better** is the server performance.

In this test, the throughput of Google server is 1,491.193/minute. It means Google server can handle 1,491.193 requests per minute. This value is quiet high so we can conclude that Google server has good performance

The **deviation** is shown in red - it indicates the deviation from the average. The **smaller** the **better**.

Let compare the performance of Google server to other web server. This is the performance test result of website http://www.yahoo.com/ (You can choose other website)



The throughput of website under test http://www.yahoo.com is 867.326/minutes. It means this server handle 867.326 requests per minute, lower than Google.

The deviation is 2689, much higher than Google (577). So we can determine the performance of  this website is less than Google server.