Performance Test Tools

Introduction to JMeter

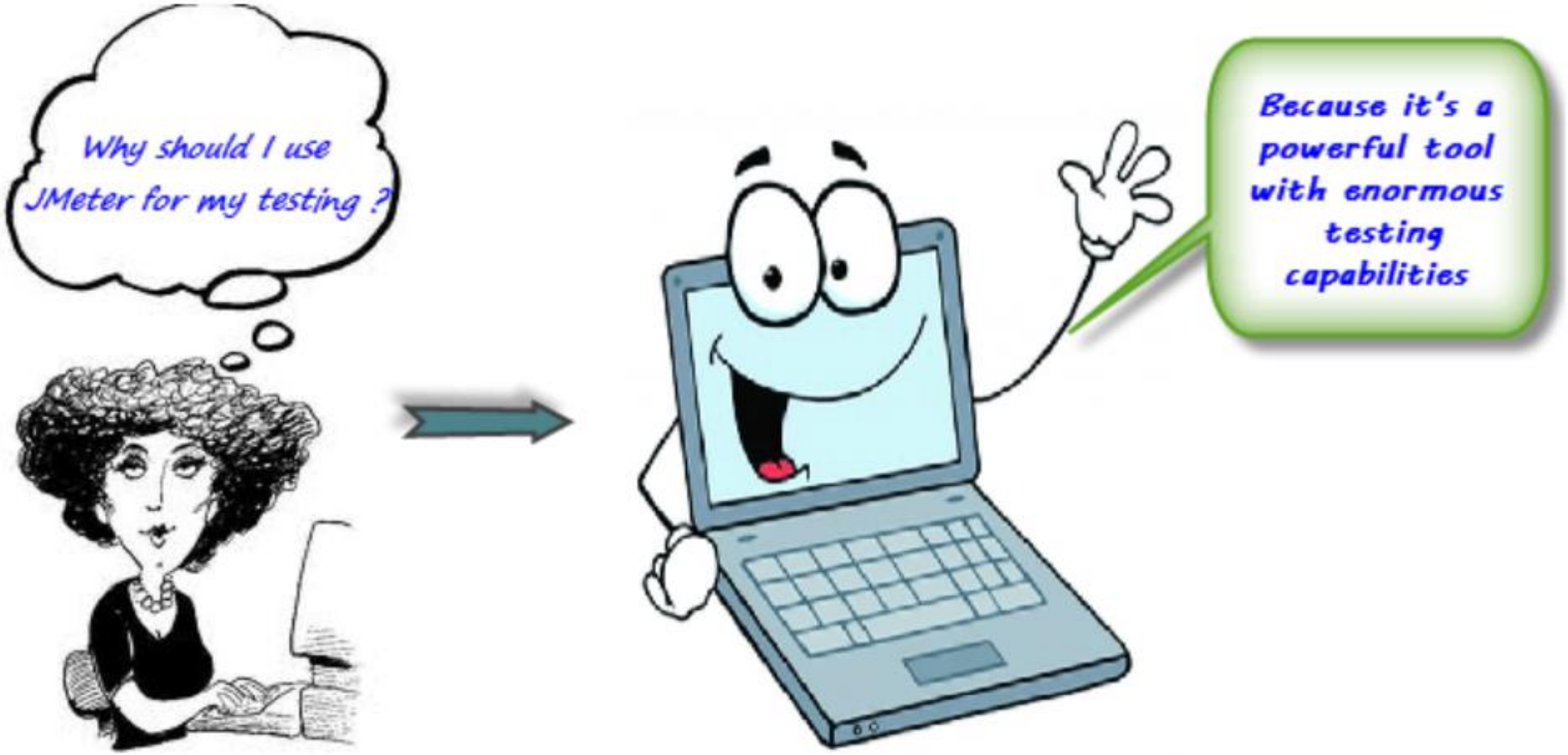
Contents

Introduction

The **Apache JMeter™** is pure Java **open source** software, which was first developed by Stefano Mazzocchi of the Apache Software Foundation, designed to load test functional behavior and measure performance.

Apache Jmeter is a popular open source performance testing tool.

Why JMeter?


Why should I use JMeter for my testing ?
Because it's a powerful tool with enormous testing capabilities

Jmeter Advantages

Open source license	Friendly GUI	Platform independent
Full multi-threading framework	Visualize Test Result	Easy installation
Highly extensible	Unlimited testing capabilities	Support multi protocol

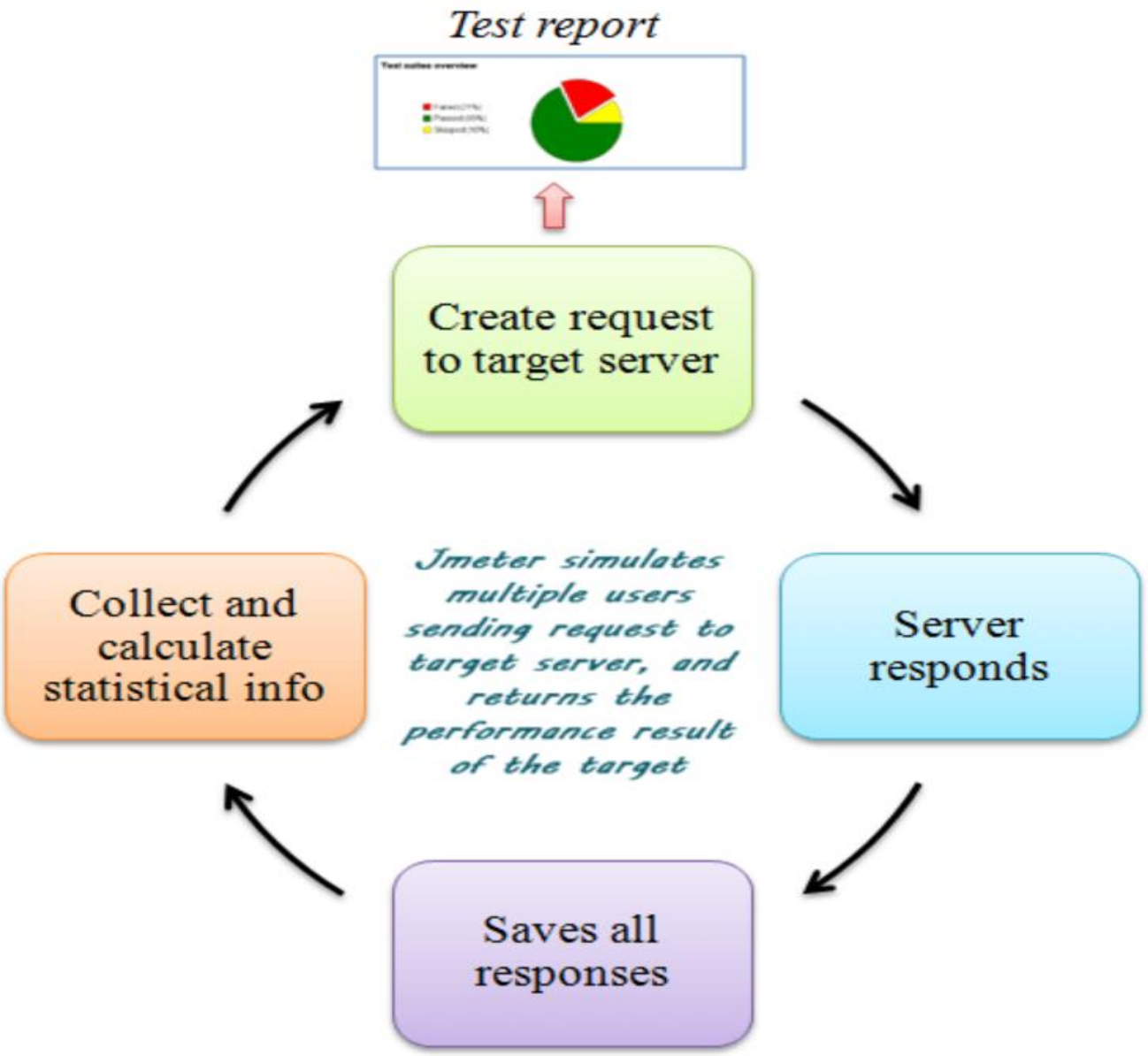
JMeter Advantages

- ✓ **Open source license**: JMeter is totally free, allows developer use the source code for the development
- ✓ **Friendly GUI**: JMeter is extremely easy to use and doesn't take time to get familiar with it
- ✓ **Platform independent**: JMeter is 100% pure Java desktop application. So it can run on multiple platforms
- ✓ **Full multithreading framework**. JMeter allows concurrent and simultaneous sampling of different functions by a separate thread group
- ✓ **Visualize Test Result:** Test result can be displayed in a different format such as chart, table, tree and log file
- ✓ **Easy installation**: You just copy and run the *.bat file to run JMeter. No installation needed.

JMeter Advantages (continue)

- ✓ **Highly Extensible**: You can write your own tests. JMeter also supports visualization plugins allow you to extend your testing
- ✓ **Multiple testing strategy**: JMeter supports many testing strategies such as Load Testing, Distributed Testing, and Functional Testing.
- ✓ **Simulation**: JMeter can simulate multiple users with concurrent threads, create a heavy load against web application under test
- ✓ **Support multi-protocol**: JMeter does not only support web application testing but also evaluate database server performance. All basic protocols such as HTTP, JDBC, LDAP, SOAP, JMS, and FTP are supported by JMeter
- ✓ **Record & Playback - Record** the user activity on the browser and simulate them in a web application using JMeter
- ✓ **Script Test**: Jmeter can be integrated with Bean Shell & Selenium for automated testing

How JMeter Works



OS Support For JMeter

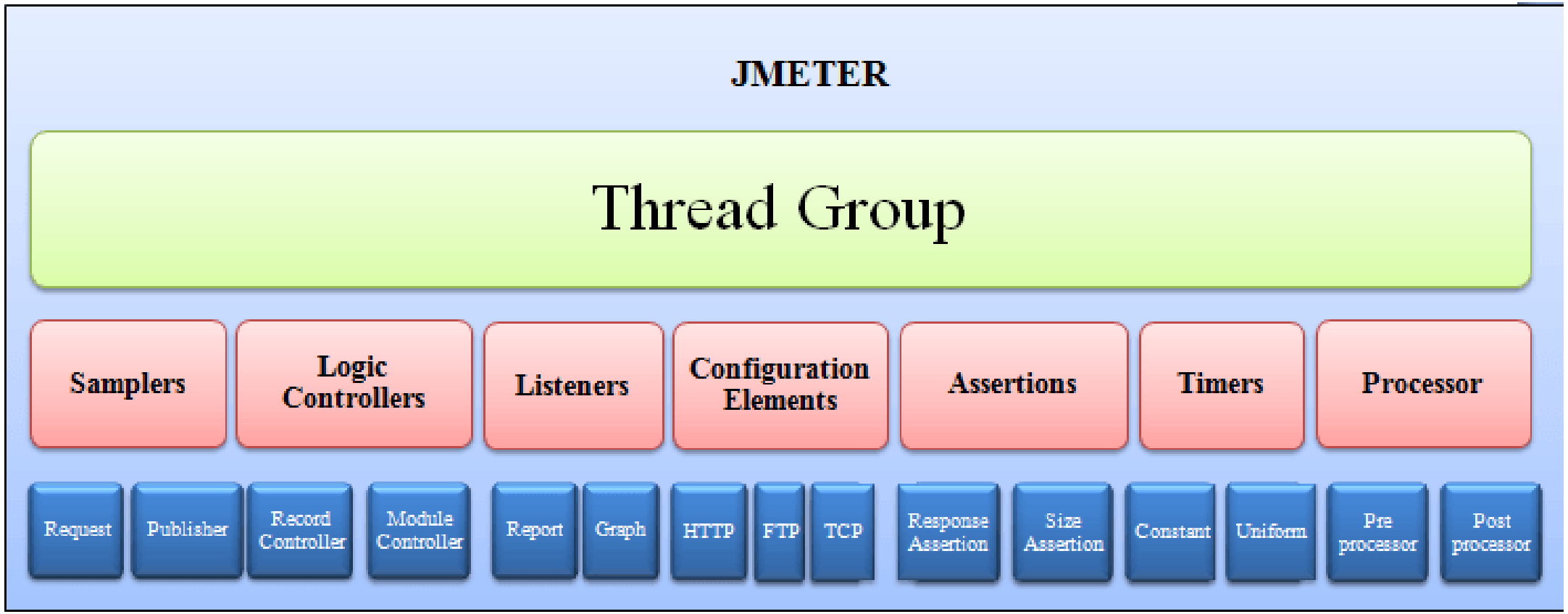
JMeter is a **pure Java** application and should run correctly on any system that has a compatible Java implementation.

Here is the list of an operating system compatible with JMeter

- Linux
- Windows
- Mac OS
- Ubuntu

JMeter Elements: Thread Group, Samplers, Listeners, Configuration

What is Element in JMeter?

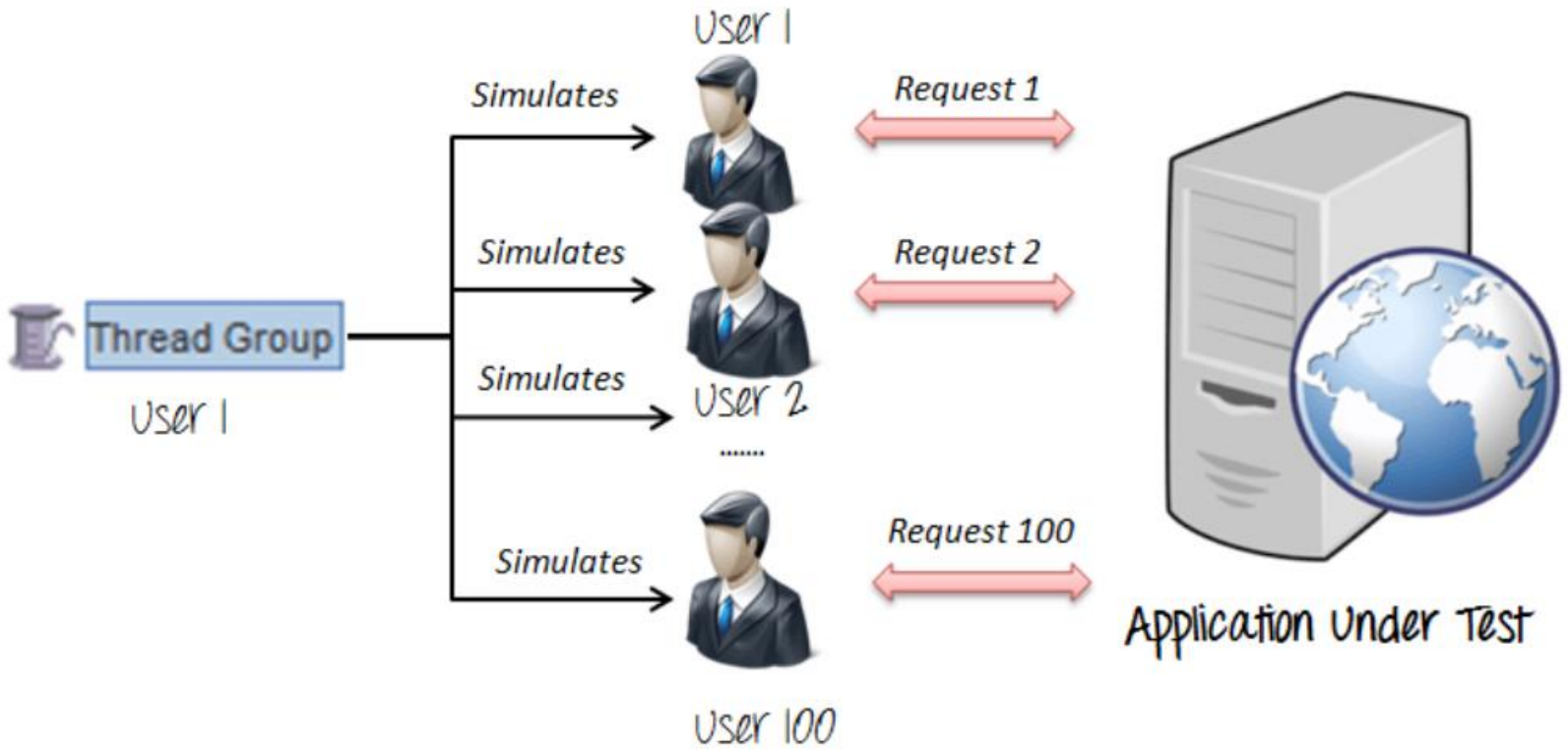


Thread Group

Thread Groups is a collection of Threads. Each thread represents one user using the application under test. Basically, each Thread simulates one real user request to the server.

The controls for a thread group allow you to Set the number of threads for each group.

For example, if you set the number of threads as 100; JMeter will create and simulate 100 user requests to the server under test



Samplers

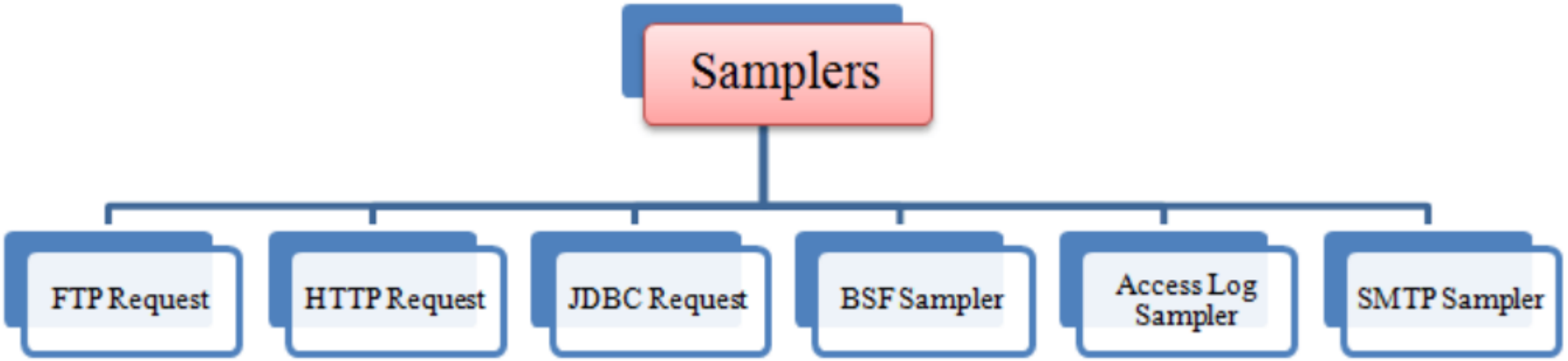
As we know already that JMeter supports testing HTTP, FTP, JDBC and many other protocols.

We already know that Thread Groups simulate user request to the server

But how does a Thread Group know which type of requests (HTTP, FTP etc.) it needs to make?

The answer is Samplers

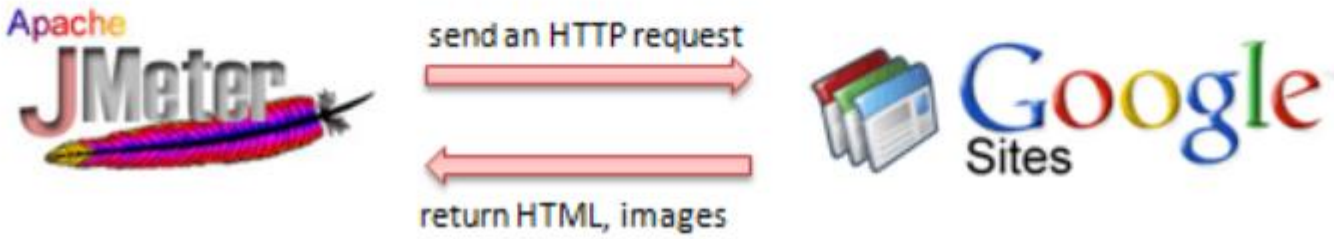
The user request could be FTP Request, HTTP Request, JDBC Request...Etc.



HTTP request:

This sampler lets you send an HTTP/HTTPS request to a web server.

Consider the example below. JMeter sends an HTTP request to Google website and retrieve HTML files or image from this website.



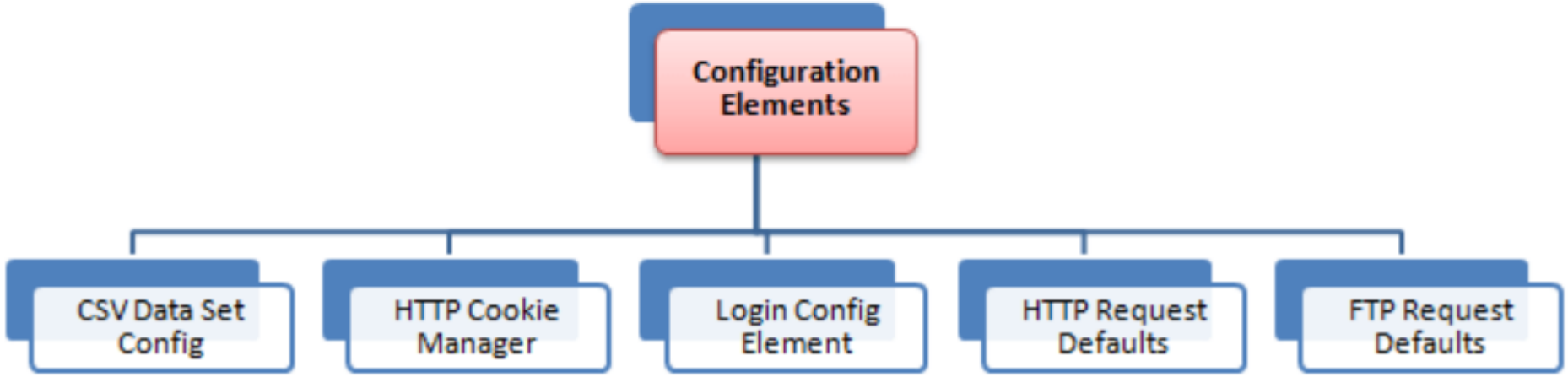
Listeners

Listeners: shows the results of the test execution. They can show results in a different format such as a tree, table, graph or log file

Configuration Elements

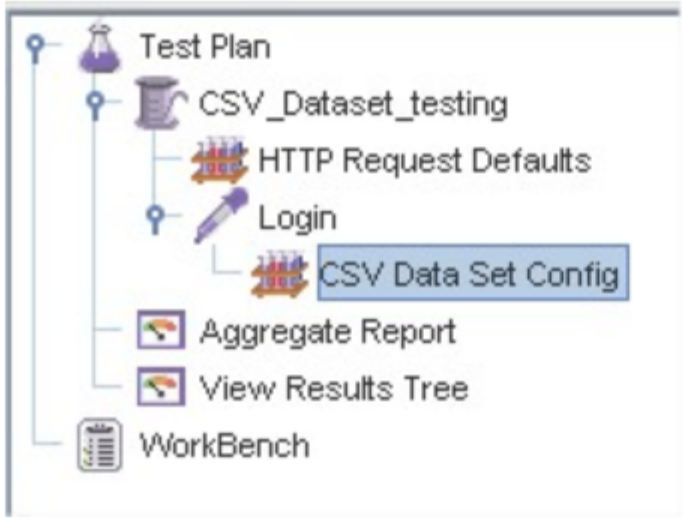
set up defaults and variables for later use by samplers.

The figure below shows some commonly used configuration elements in JMeter

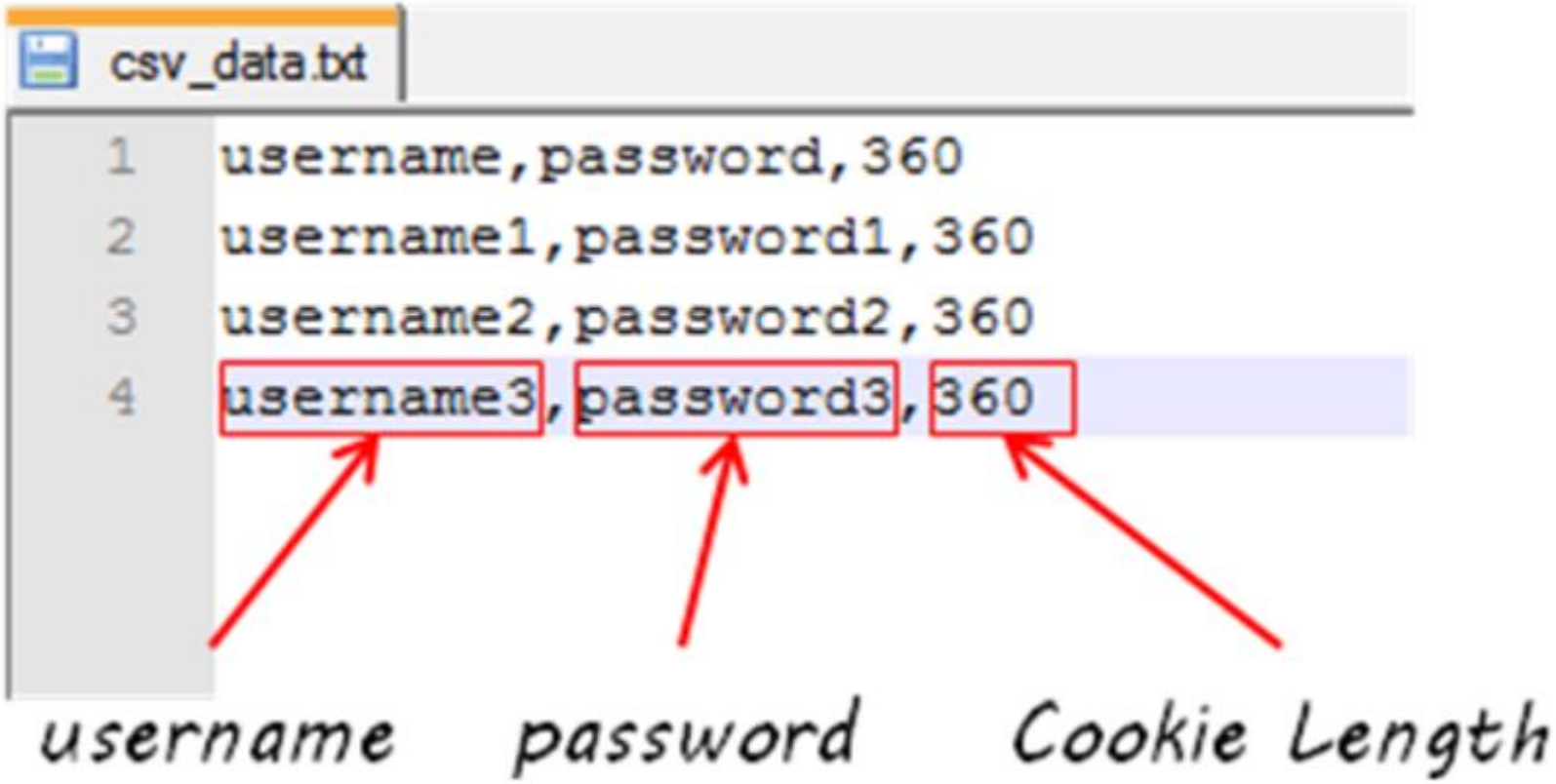


CSV Data Set Config:

Suppose you want to test a website for 100 users signing-in with different credentials. You do not need to record the script 100 times! You can parameterization the script to enter different login credentials. This login information (e.g. Username, password) could be stored in a text file. JMeter has an element that allows you to read different parameters from that text file. It is "CSV Data Set Config", which is used to read lines from a file, and split them into variables.



This is an example of CSV Data. It's a text file which contains user and password to login your target website



HTTP request default

This element lets you set default values that your HTTP Request controllers use.

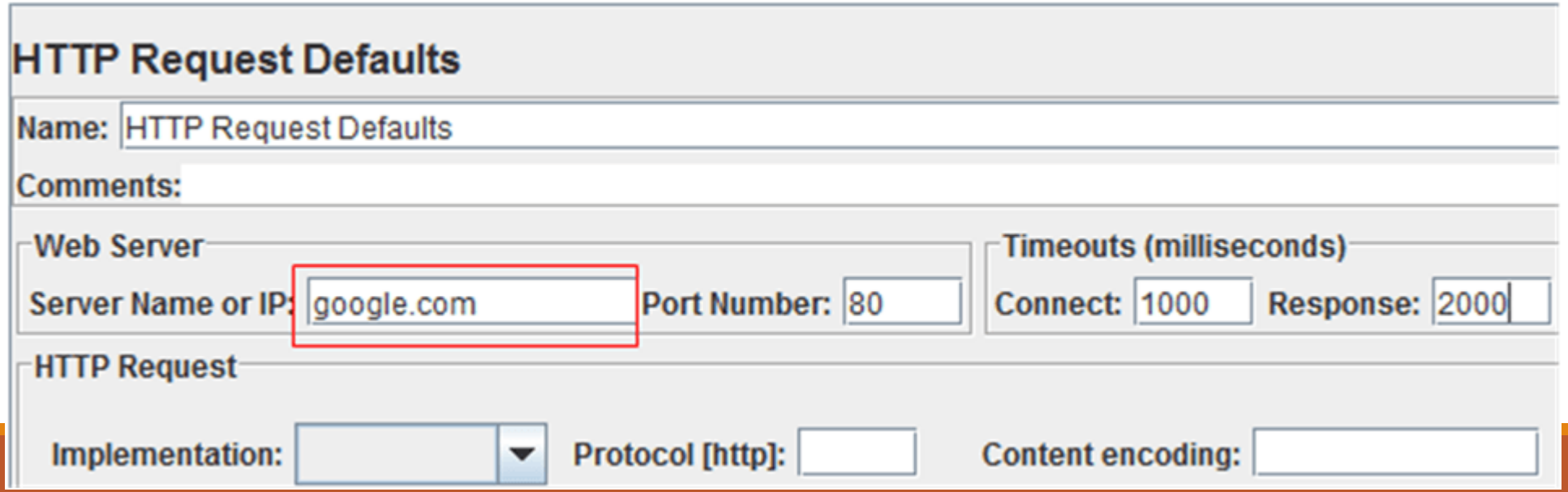
For example,

You are sending 100 HTTP requests to the server google.com

You would have to manually enter server name = google.com for all these 100 requests

Instead, you could add a single HTTP request defaults with the "Server Name or IP" field = google.com

No need to type 100 times!

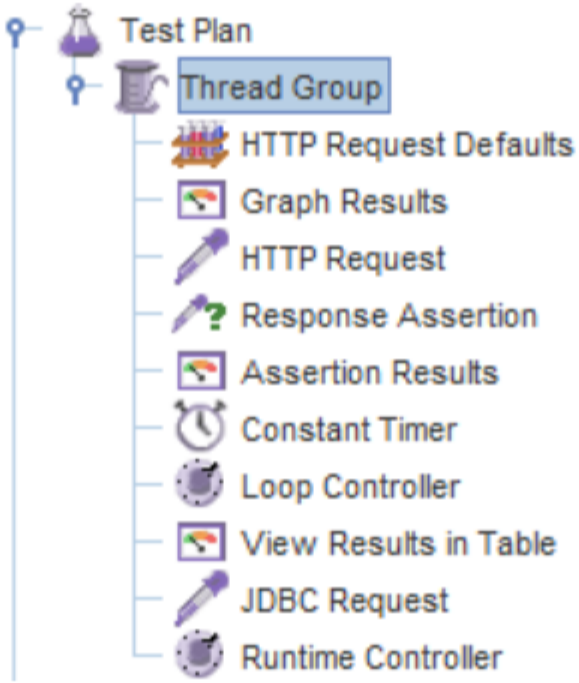


What is a Test Plan?

Test Plan is where you add elements required for your JMeter Test.

It stores all the elements (like ThreadGroup, Timers etc) and their corresponding settings required to run your desired Tests.

The following figure shows an example of Test Plan



How to add Elements?

Adding Elements is the **essential** step to build a Test Plan because without adding elements, JMeter **cannot** execute your Test Plan

A Test Plan includes many Elements such as Listener, Controller, and Timer

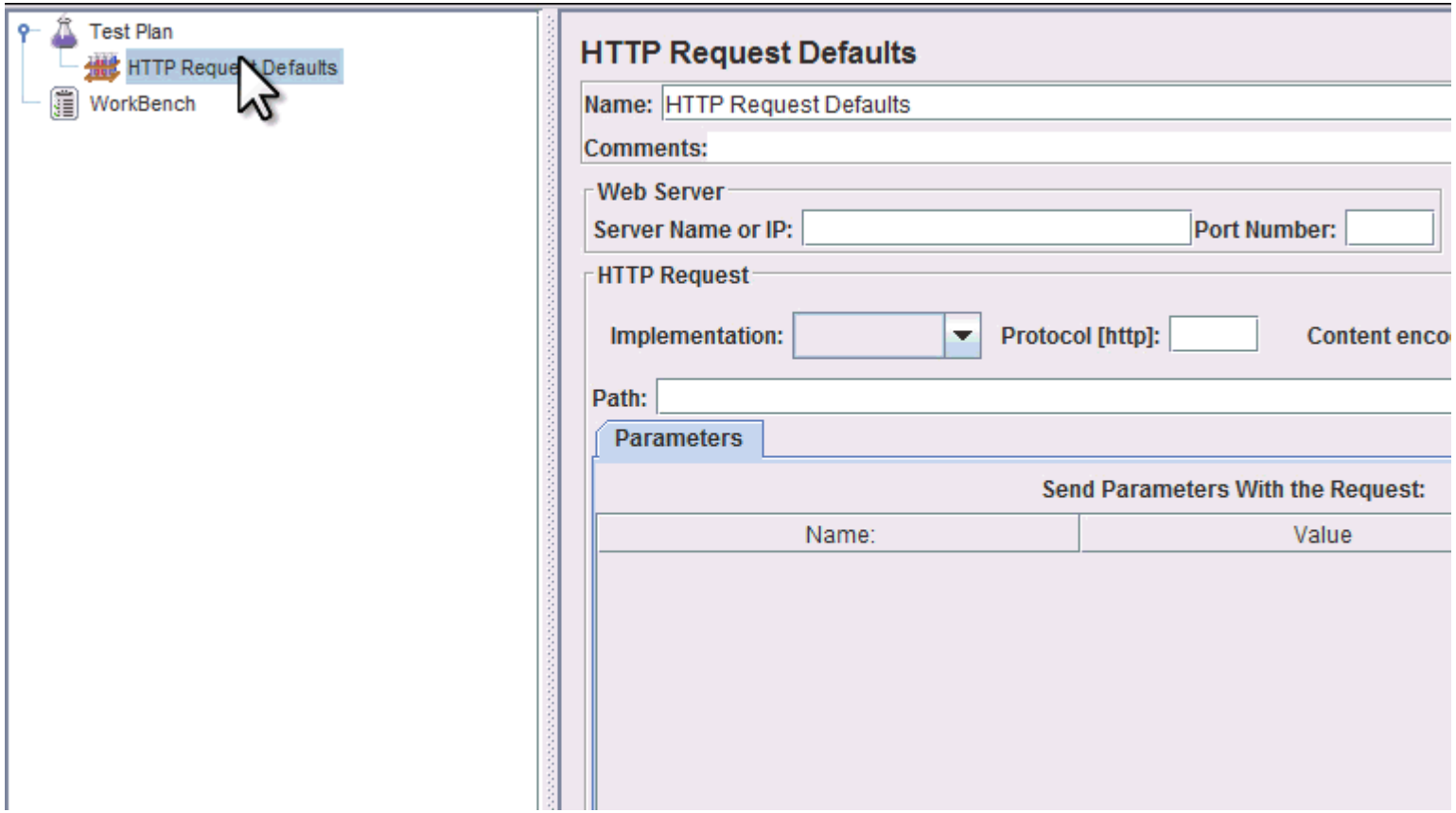
You can add an element to test plan by right-clicking on a **Test Plan** and choose new elements from "**Add**" list.

Suppose, you want to add 2 elements to Test Plan **BeanShell Assertion** and **Java Request Default**

- Right click **Test Plan -> Add -> Assertion -> Bean Shell Assertion**
- Right click **Test Plan -> Add -> Config Element -> Java Request Default**

You can also **remove** an unused element

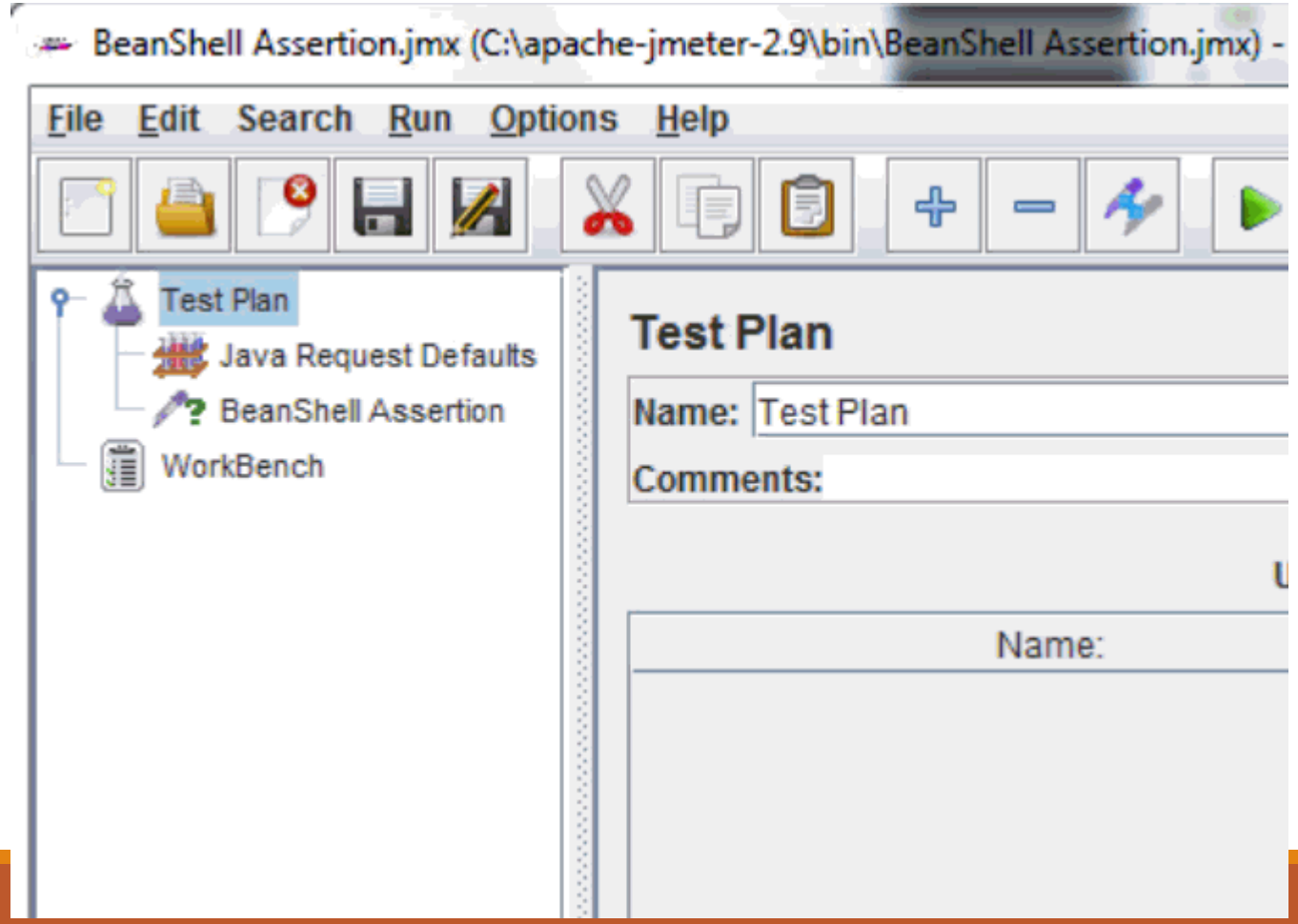
Let's say, you want to remove element "**HTTP Request Defaults**", select "HTTP Request Default" -> Right click-> choose **Remove** from the context menu -> Click **Yes** to confirm delete this element on message box



How to Save a Test Plan

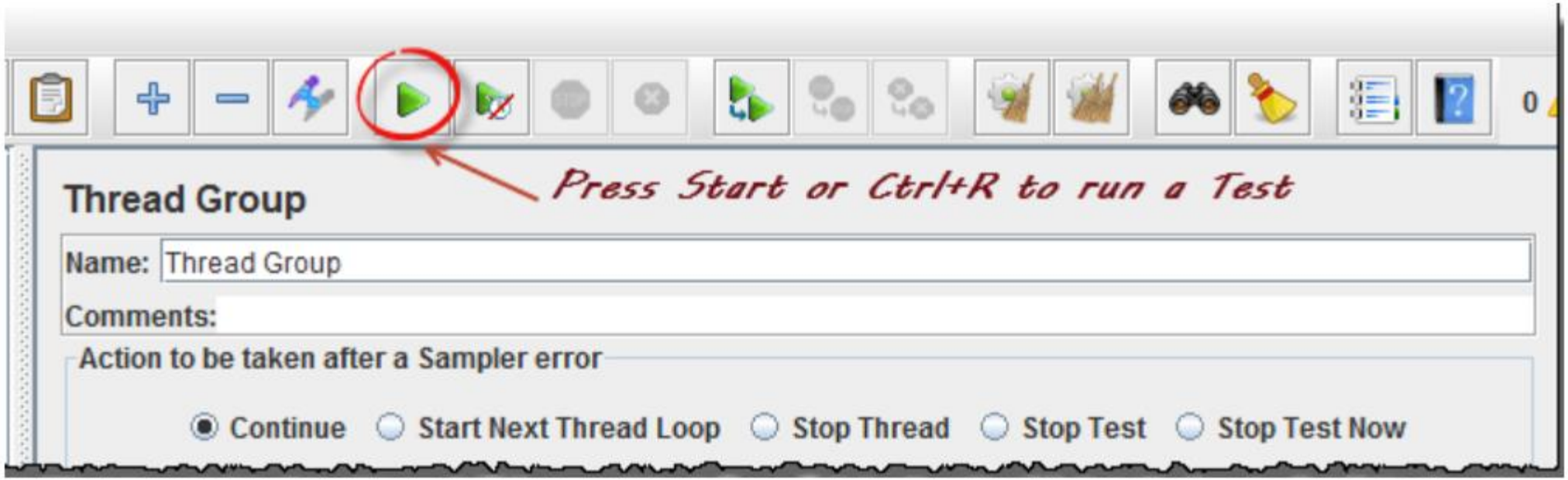
Before running a test, you should save your Test Plan first. Saving your Test Plan helps you avoid unexpected error when running the test plan. Steps to saving Test plan -

1. ***File -> Save Test Plan as-> a Dialog box display***
2. *Enter a filename of Test Plan ->click* ***Save***



How to Run Test Plan

To run your single or multiple test plans, choose **Start** (Control + R) from the **Run** menu item.



When JMeter is running, it shows a small green box at the right-hand end of the menu bar.

The numbers to the left of the green box are the number of **active threads** / **total number** of threads.

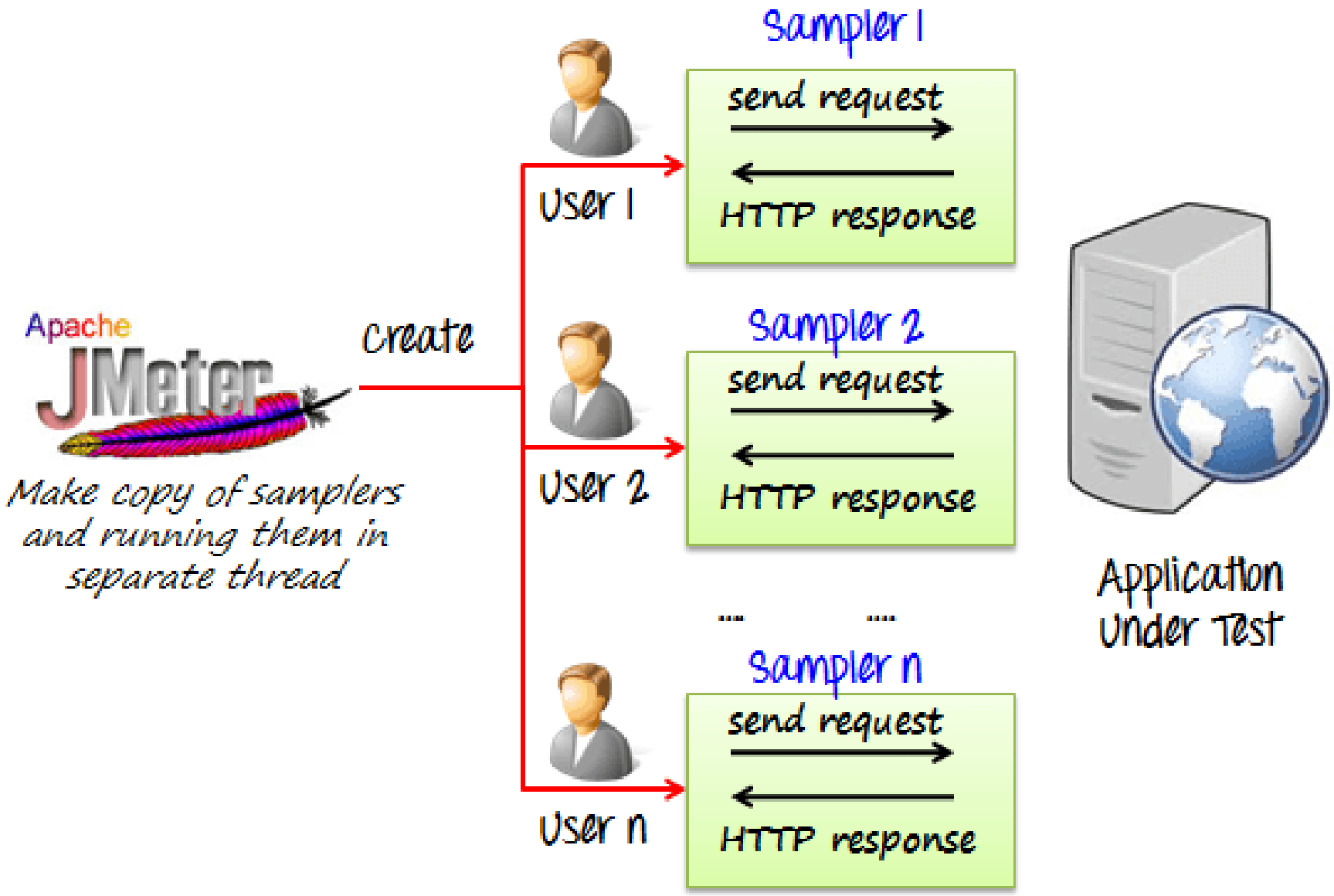
To Stop the Test, press **Stop** button or use short key Ctrl + '.'

Test Report

When test execution is done, you can get the test report. The test report includes the error log file, which is saved in jmeter.log, and the test results summary. Here is a sample log file of JMeter

> - 2013/08/18 08:41:12 INFO - jmeter.JMeter: Copyright (c) 1998-2013 The Apache Software Foundation
> - 2013/08/18 08:41:12 INFO - jmeter.JMeter: Version 2.9 r1437961
> - 2013/08/18 08:41:12 INFO - jmeter.JMeter: java.version=1.7.0_25
> - 2013/08/18 08:41:12 INFO - jmeter.JMeter: java.vm.name=Java HotSpot(TM) Client VM
> - 2013/08/18 08:41:12 INFO - jmeter.JMeter: os.name=Windows 7
> - 2013/08/18 08:41:12 INFO - jmeter.JMeter: os.arch=x86
> - 2013/08/18 08:41:12 INFO - jmeter.JMeter: os.version=6.1
> - 2013/08/18 08:41:12 INFO - jmeter.JMeter: file.encoding=Cp1252
> - 2013/08/18 08:41:12 INFO - jmeter.JMeter: Default Locale=English (United States)
> - 2013/08/18 08:41:12 INFO - jmeter.JMeter: JMeter Locale=English (United States)
> - 2013/08/18 08:41:12 INFO - jmeter.JMeter: JMeterHome=C:\Nguyen\Source_code\apache-jmeter-2.9

How to Use JMeter for Performance & Load Testing

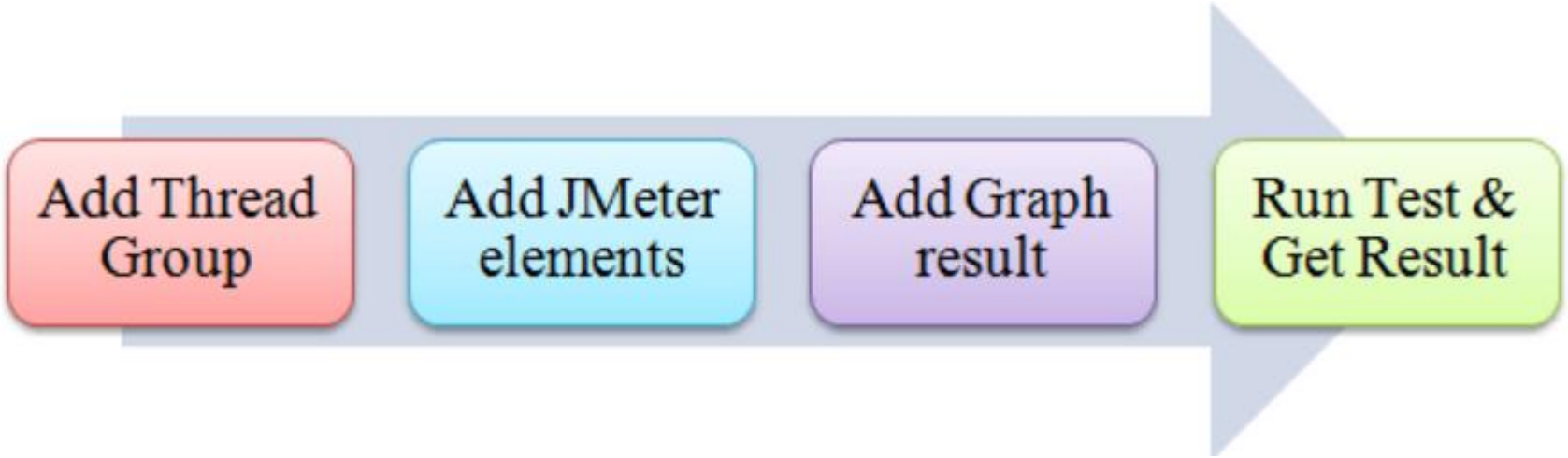


Create a Performance Test Plan in JMeter

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-

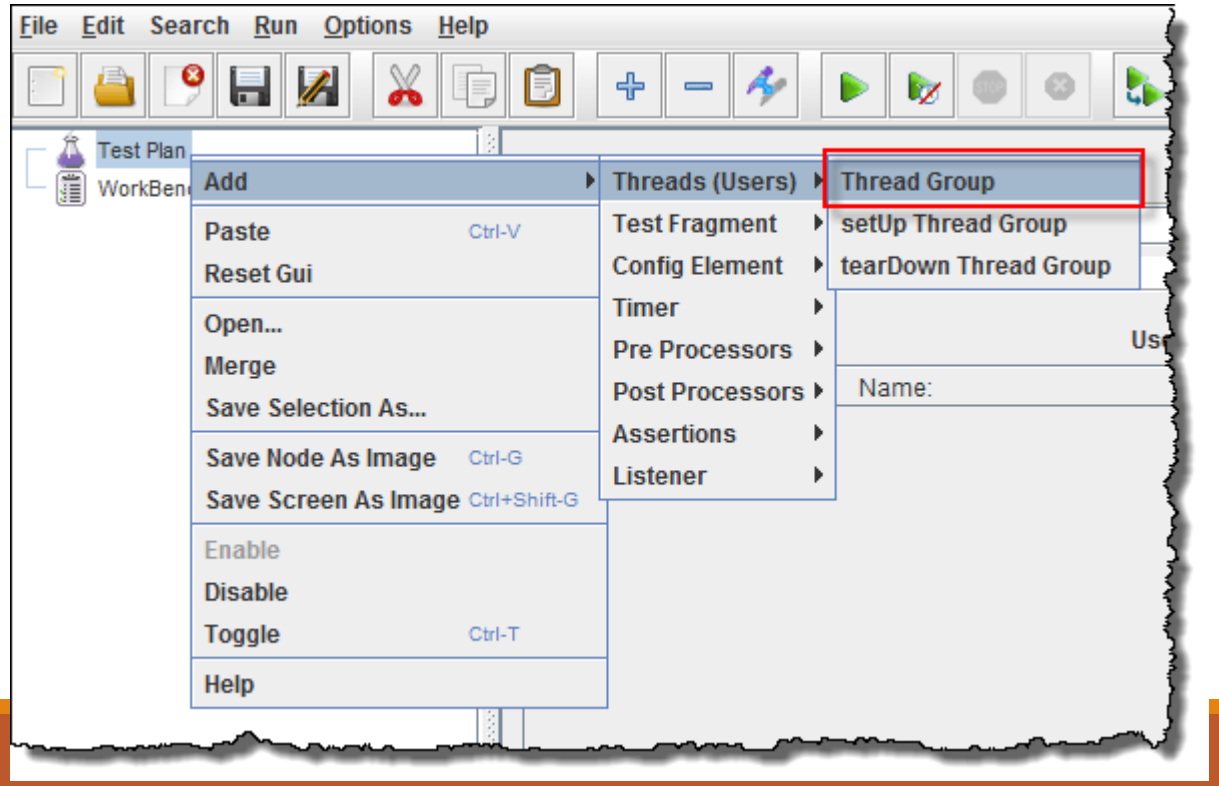
- **Normal Load**: Average number of users visit your website
- **Heavy Load**: The maximum number of users visit your website
- What is your **target** in this test?



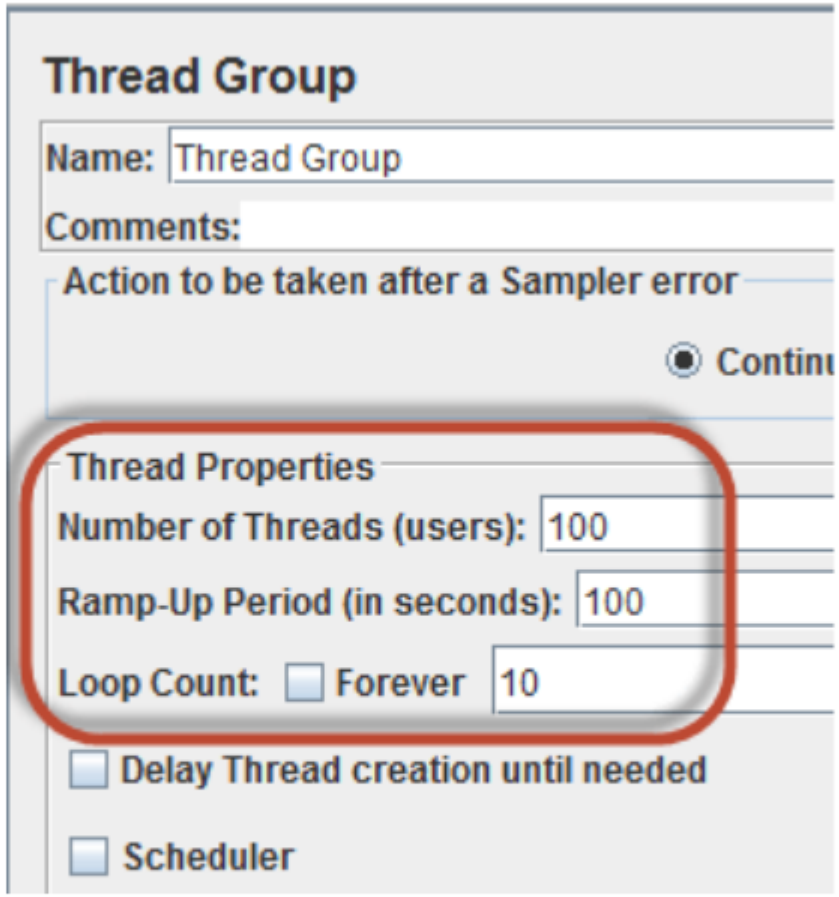
Step 1) Add Thread Group

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**

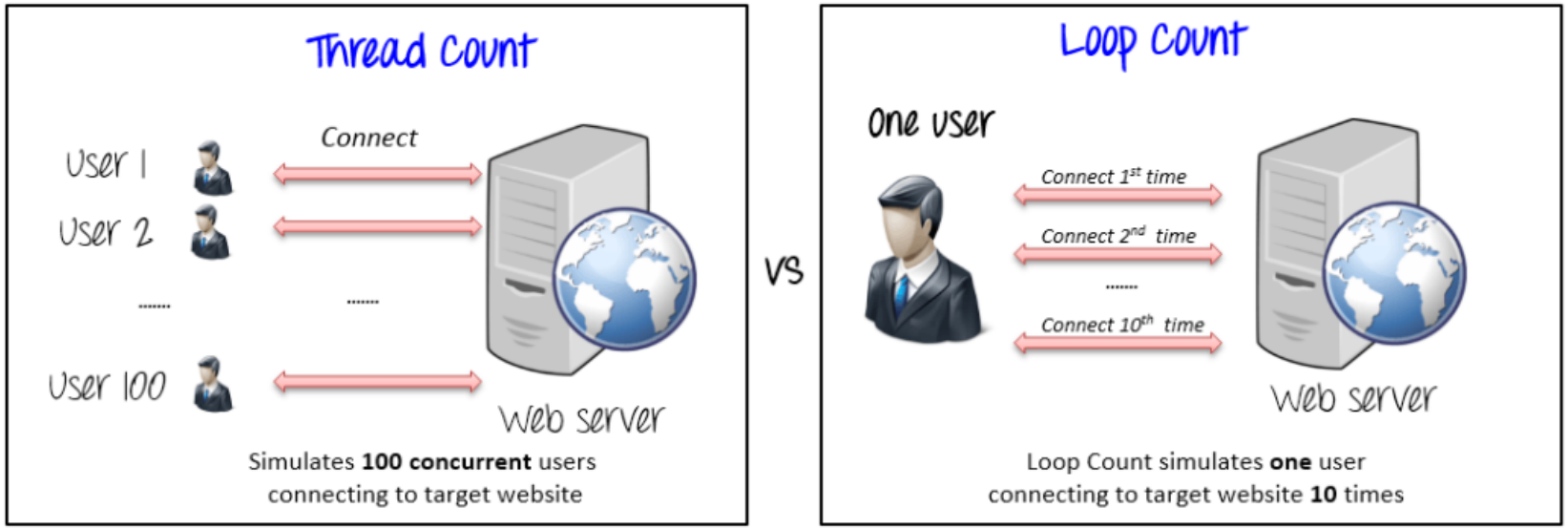


In the Thread Group control panel, enter Thread Properties as follows:

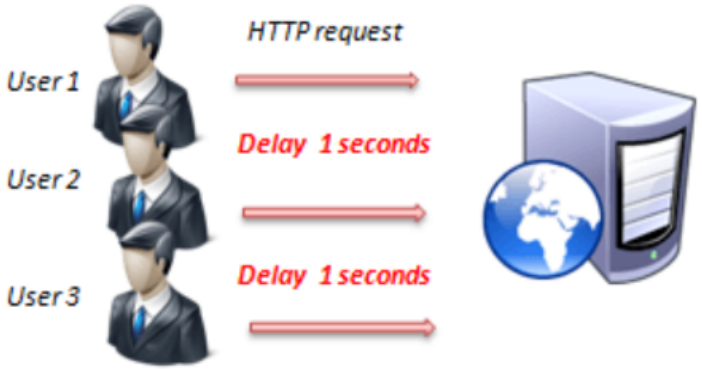


- **Number of Threads**: 100 (Number of users connects to the target website: 100)
- **Loop Count**: 10 (Number of time to execute testing)
- **Ramp-Up Period**: 100

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



Ramp-Up Period tells JMeter how long to **delay** before starting the 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 seconds /100 users)

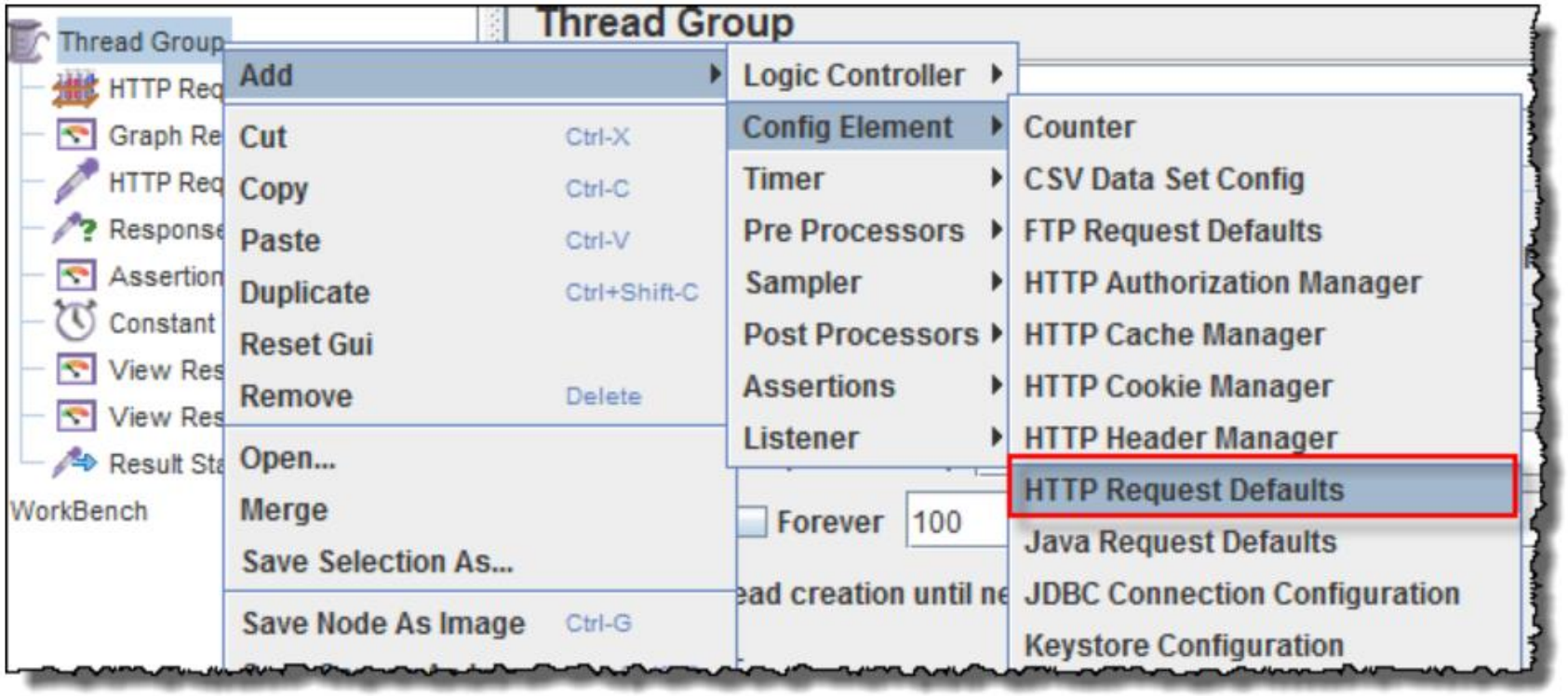


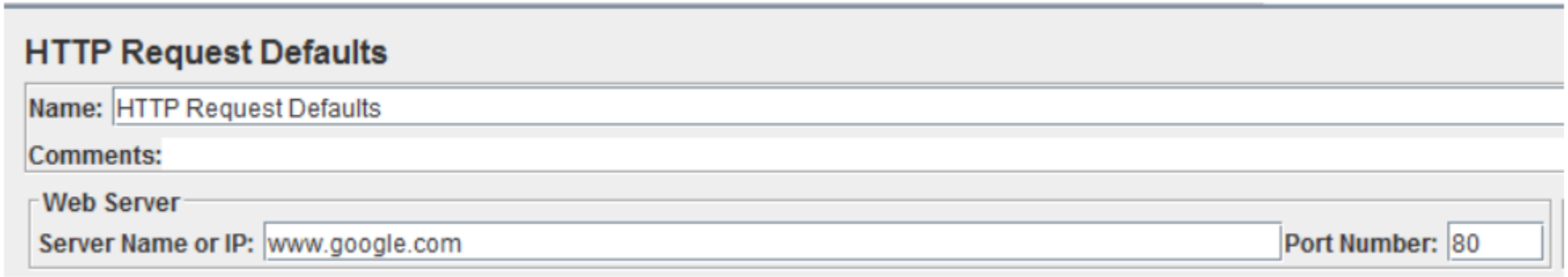
Step 2) Adding JMeter elements

Now we determine what JMeter elements in this test. The elements are

- **HTTP request Default**

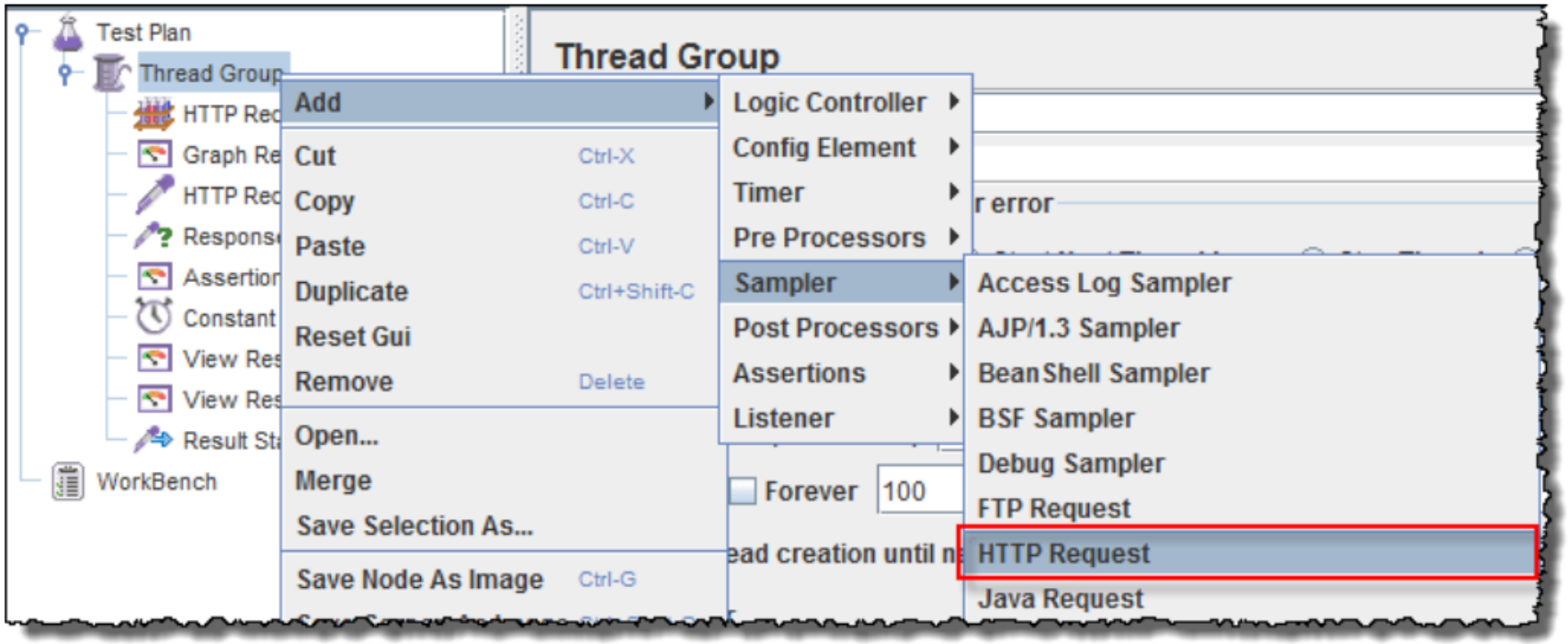
This element can be added by right-clicking on the Thread Group and selecting: **Add -> Config Element -> HTTP Request Defaults.**





- **HTTP Request**

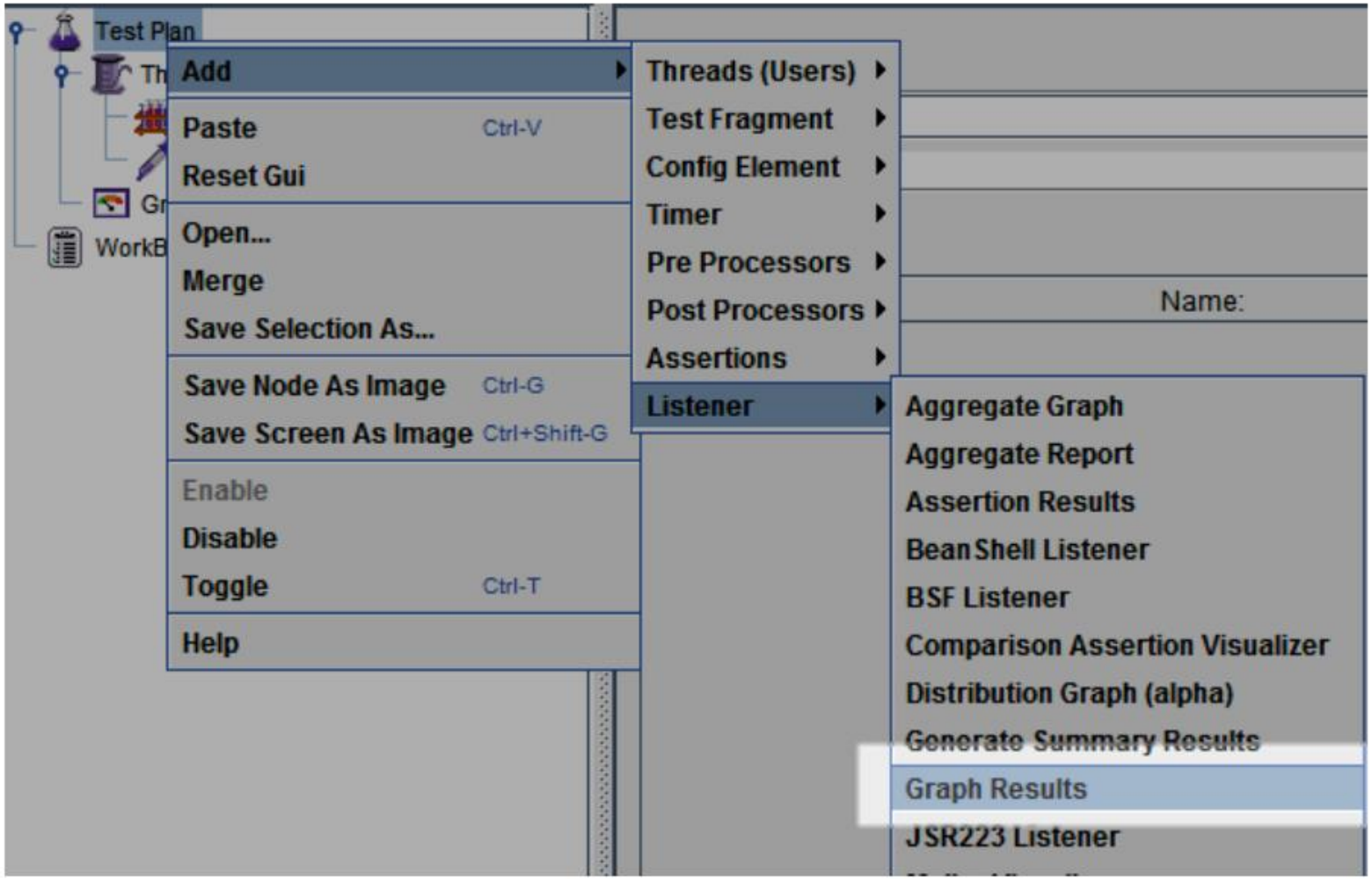
Right-click on Thread Group and select: **Add** -> **Sampler** -> **HTTP Request**.



Step 3) Adding Graph result

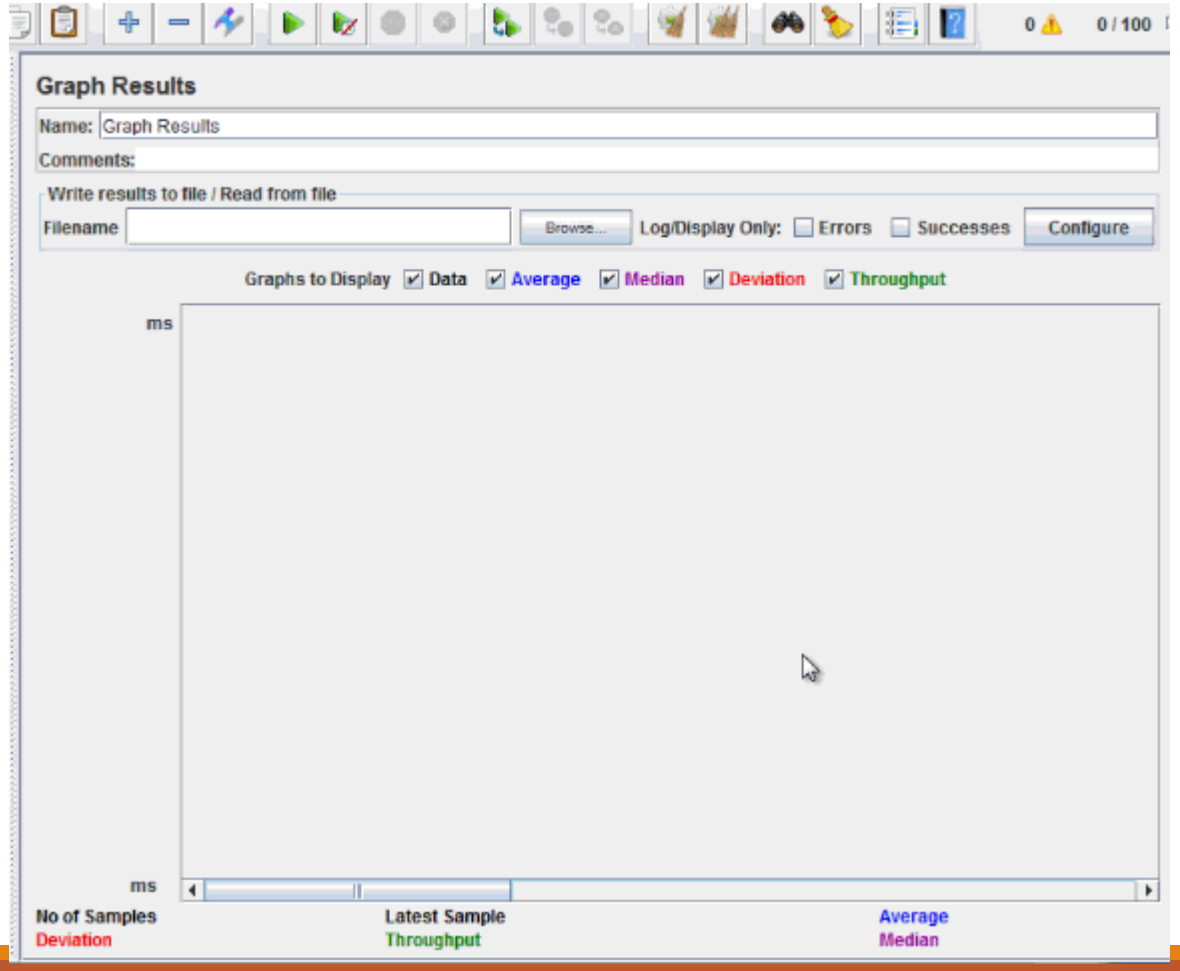
JMeter can show the test result in Graph format.

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



Step 4) Run Test and get the test result

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



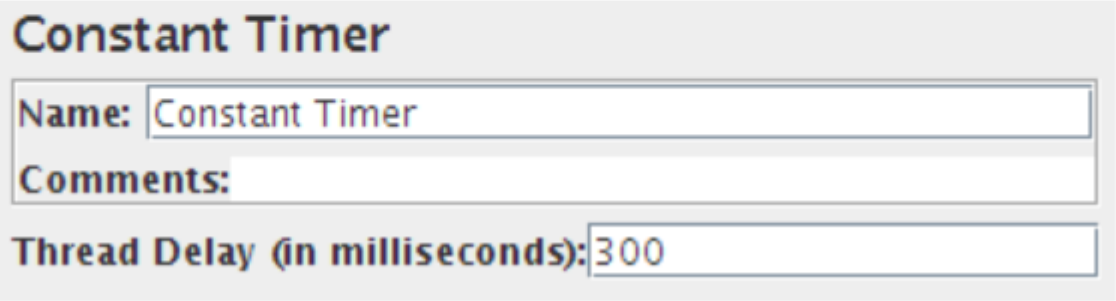
Jmeter Timers: Constant, Gaussian Random, Uniform [Example]

What are Timers?

By default, JMeter sends the request **without pausing** between each request. In that case, JMeter could **overwhelm** your test server by making too many requests in a short amount of times.

Constant Timer:

Constant timer delays each user request for the **same** amount of time.

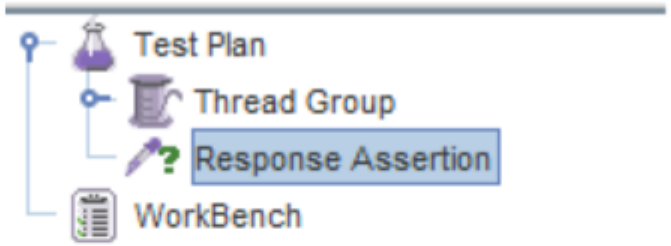


How to use Assertions in JMeter (Response Example)

What is an Assertion?

Assertion help verifies that your server under test returns the **expected** results.

Response Assertion



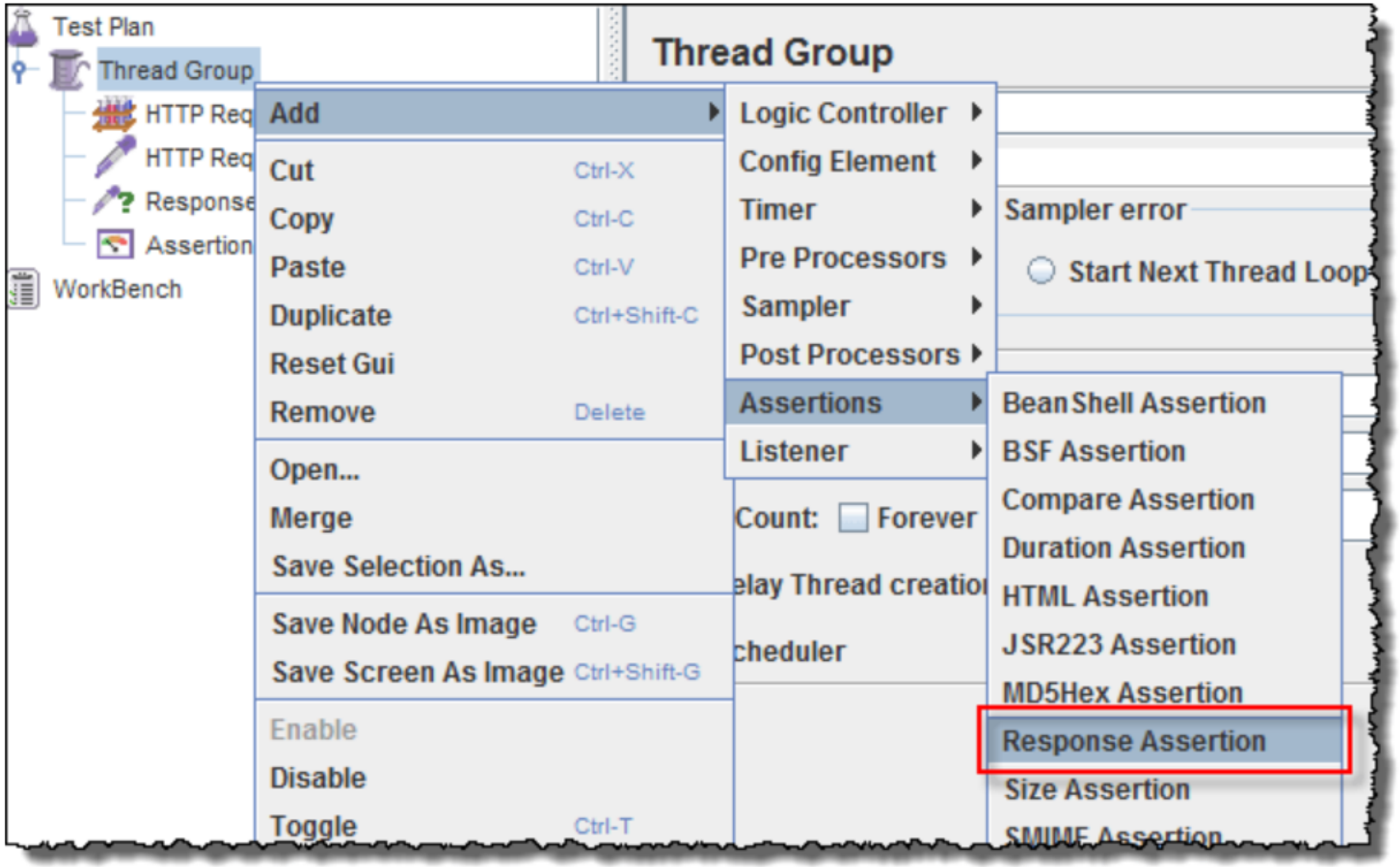
The response assertion lets you add pattern strings to be compared against various fields of the server response.

For example, you send a user request to the website http://www.google.com and get the server response. You can use Response Assertion to verify if the server response **contains** expected pattern string (e.g. "OK").

Steps to use Response Assertion

Step 1) Add Response Assertion

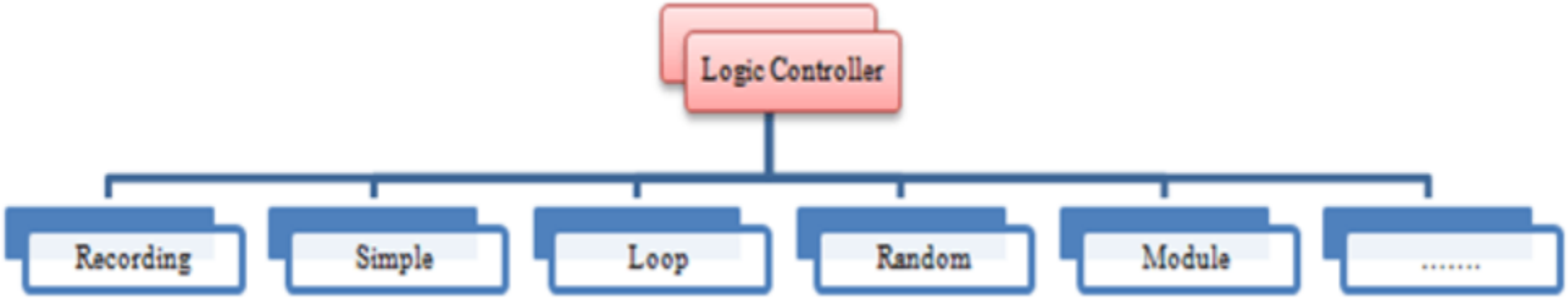
Right-Click **Thread Group -> Add -> Assertions -> Response Assertion**



Controllers in JMeter: Loop, Simple, Transaction, Module, Random

What is the Logic Controller?

Logic Controllers let you define the order of processing request in a Thread. It lets you control "when" to send a user request to a web server. For example, you can use Random Controllers to send HTTP requests to the server randomly

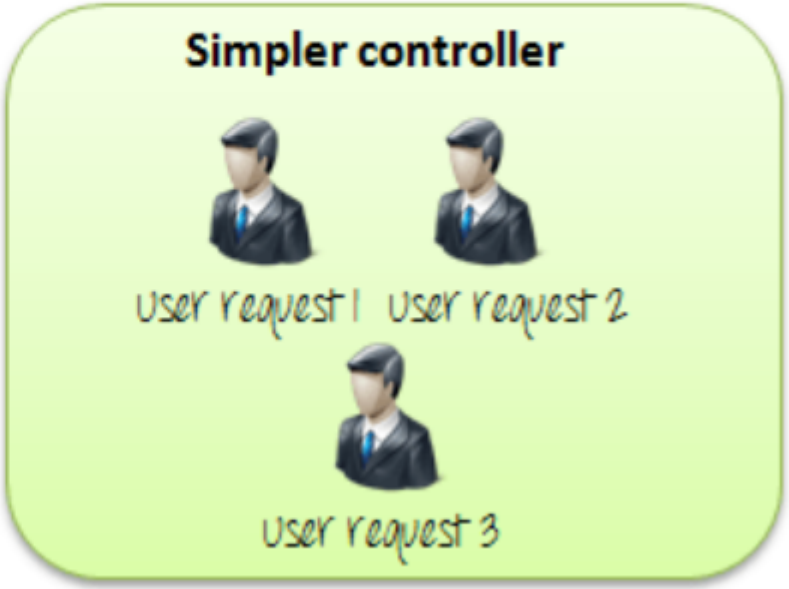


Recording Controller:

JMeter can **record** your Testing steps; a recording controller is a **placeholder** to store these recording steps.

Simple Controller:

Simple Controller is just a **container** for user request.



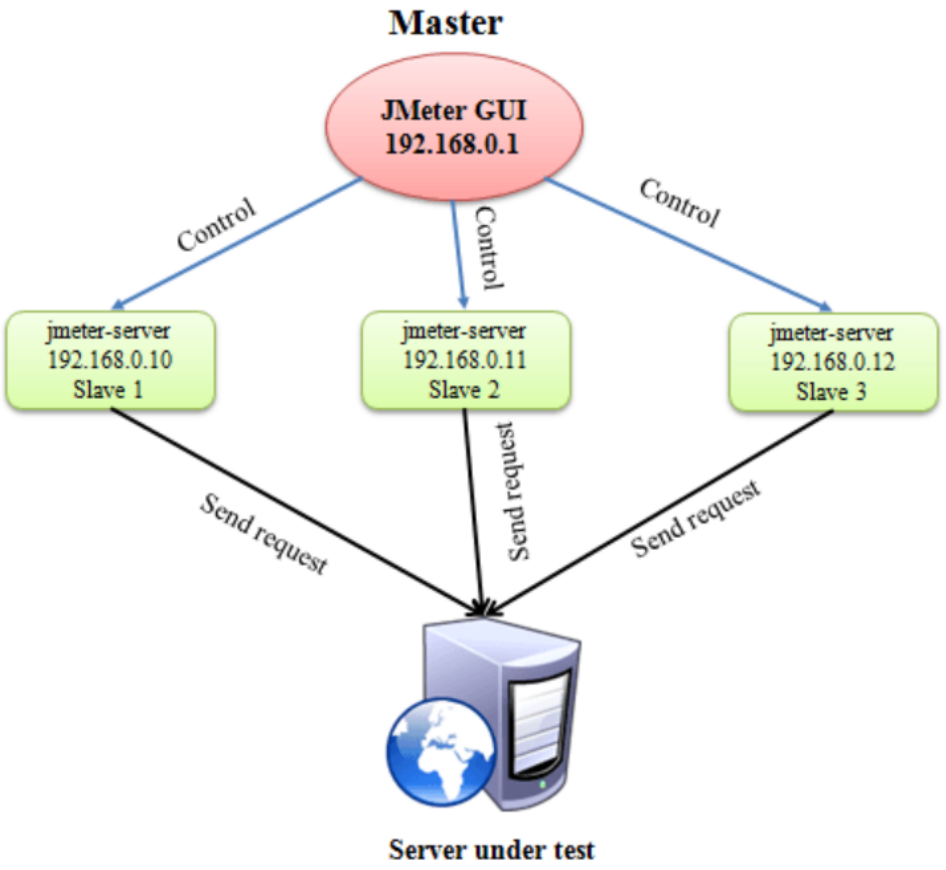
Loop Controller:

Loop Controller makes the user request run **a specified number of times** or run **forever** as shown in figure:

Advanced in JMeter

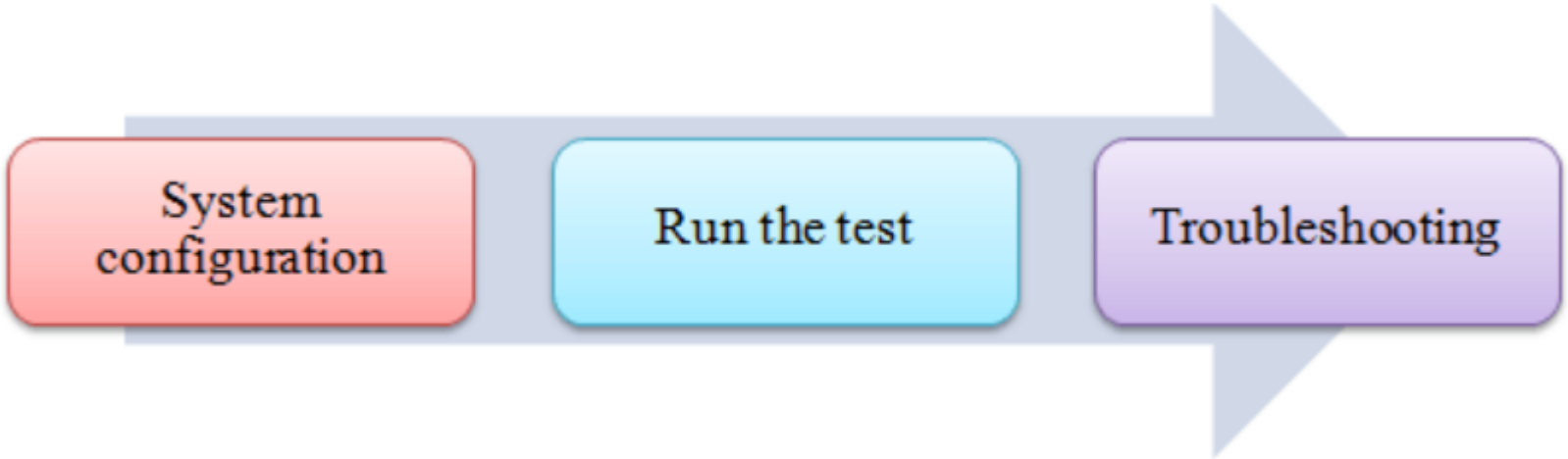
Jmeter Distributed (Remote) Testing: Master Slave Configuration

Distributes testing uses client-server model as the figure below:



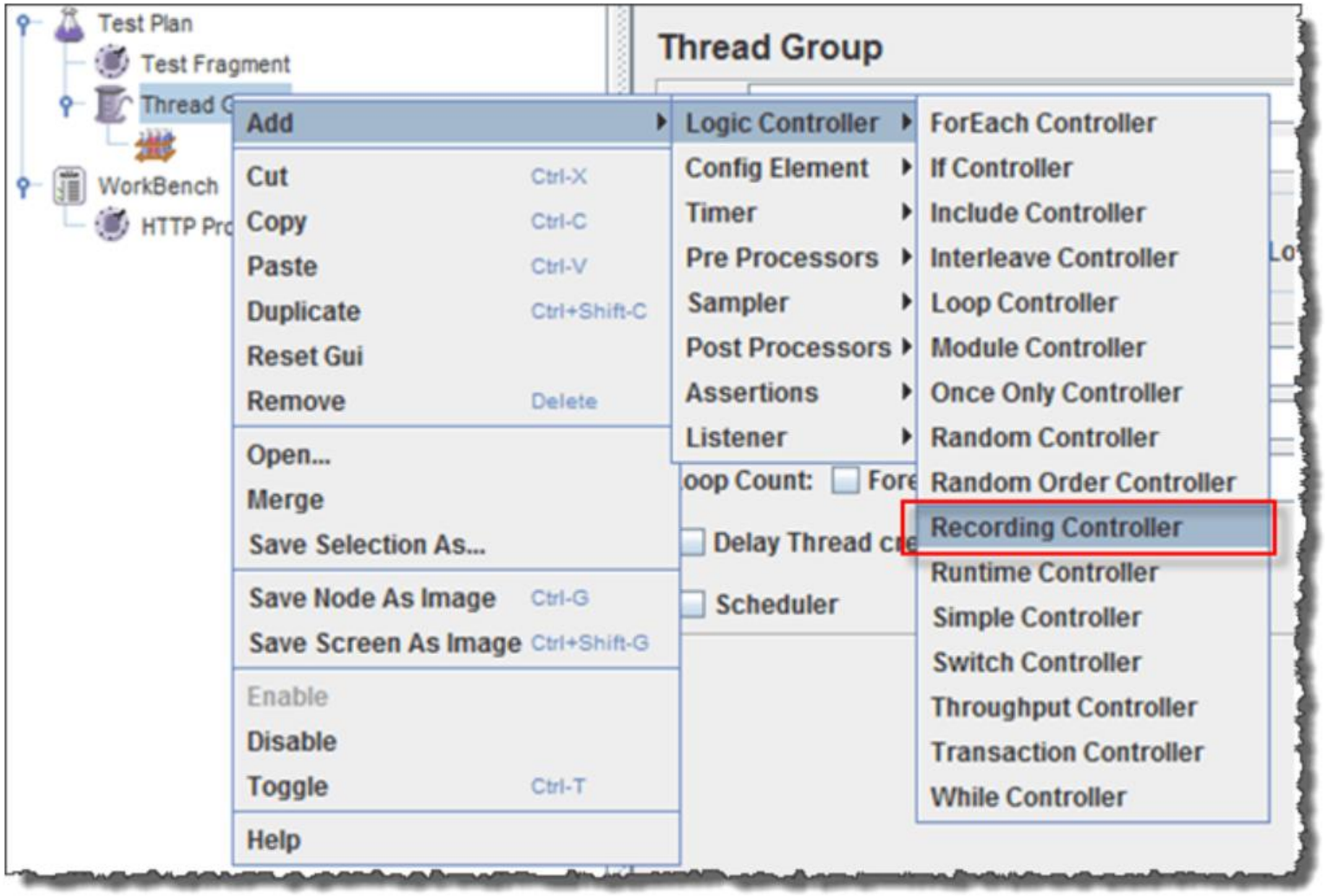
Jmeter Distributed (Remote) Testing: Master Slave Configuration

- **Master**: the system running JMeter GUI, control each slave.
- **Slave**: the system running JMeter-server, receive a command from the master and send a request to a server under test.
- **Target**: the web server under test, get a request from slaves.



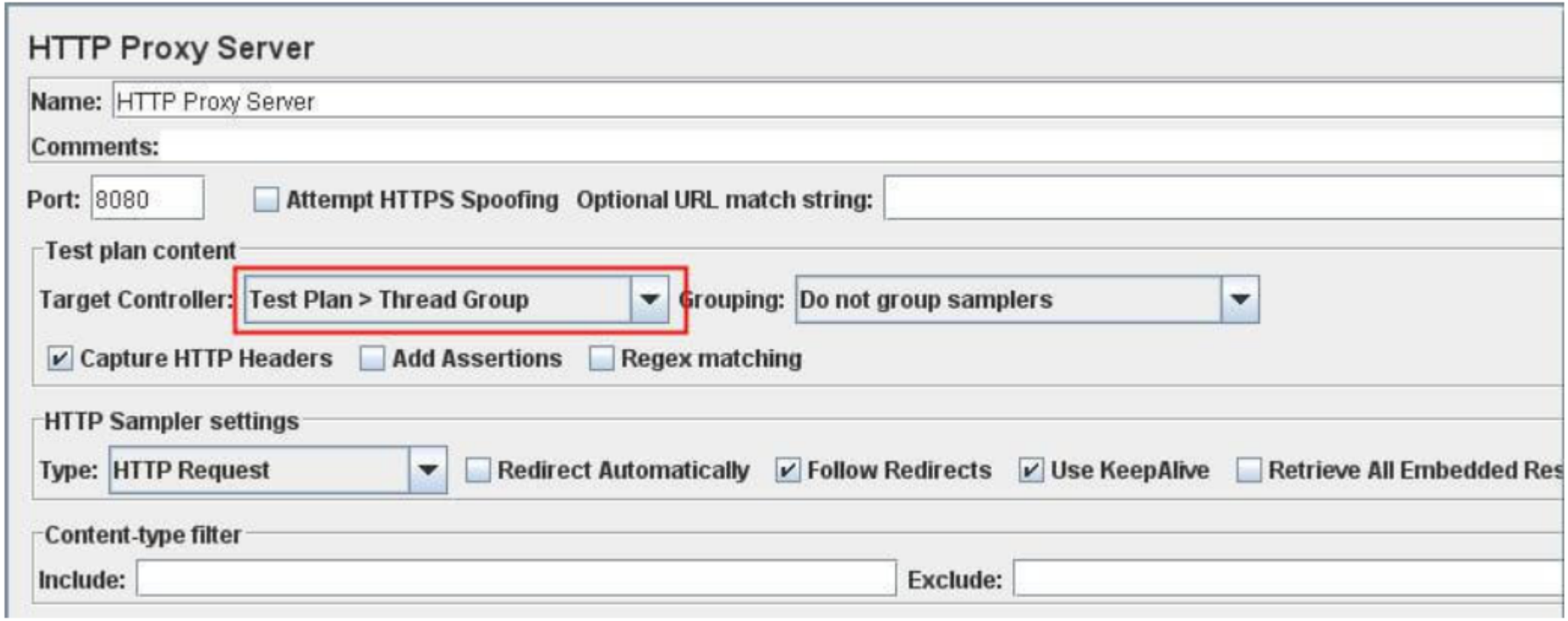
HTTP Proxy Server in JMeter: Record Example Script

Recording Controller



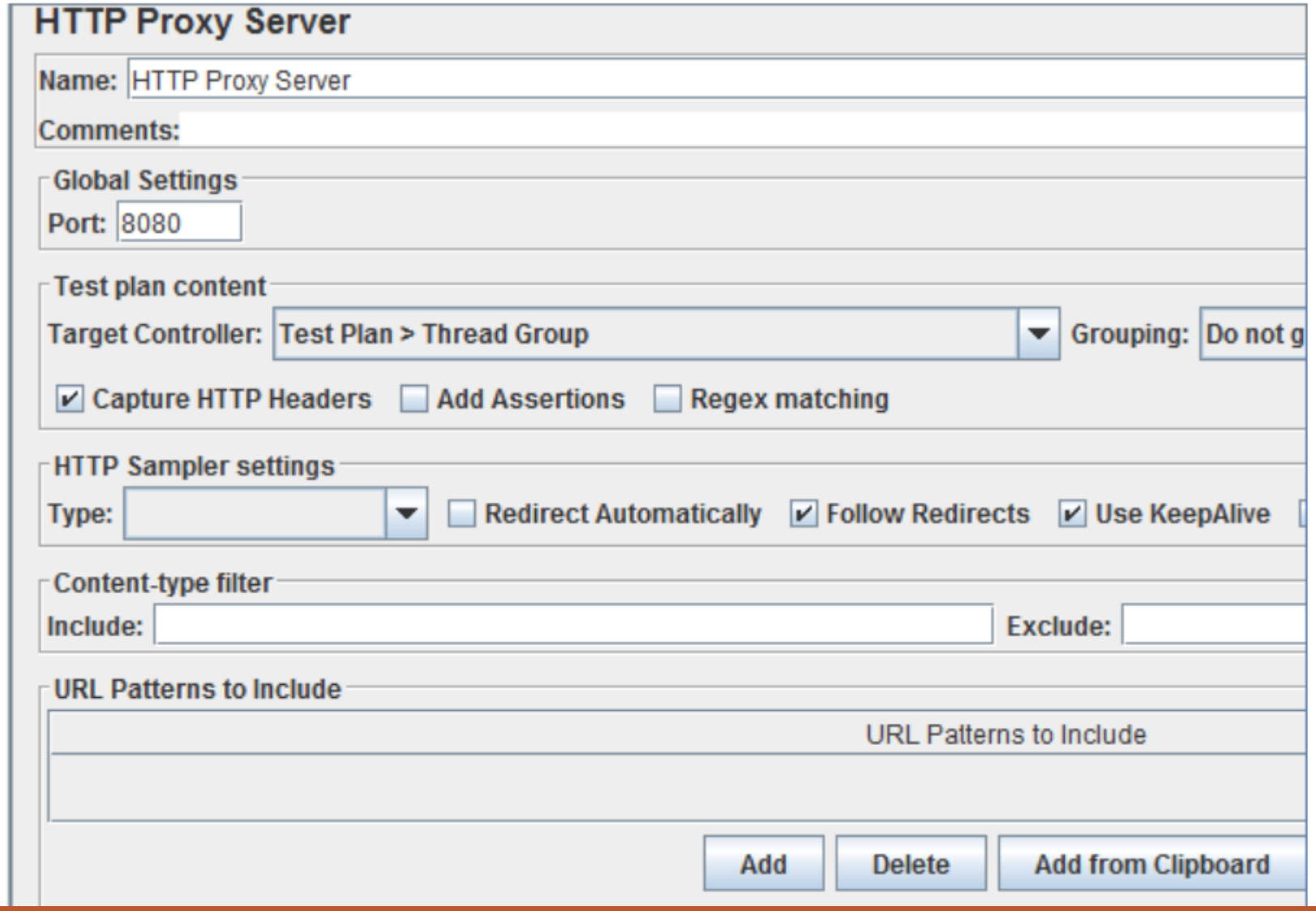
HTTP Proxy Server in JMeter: Record Example Script

Set **Target Controller** where your recorded scripts will be added

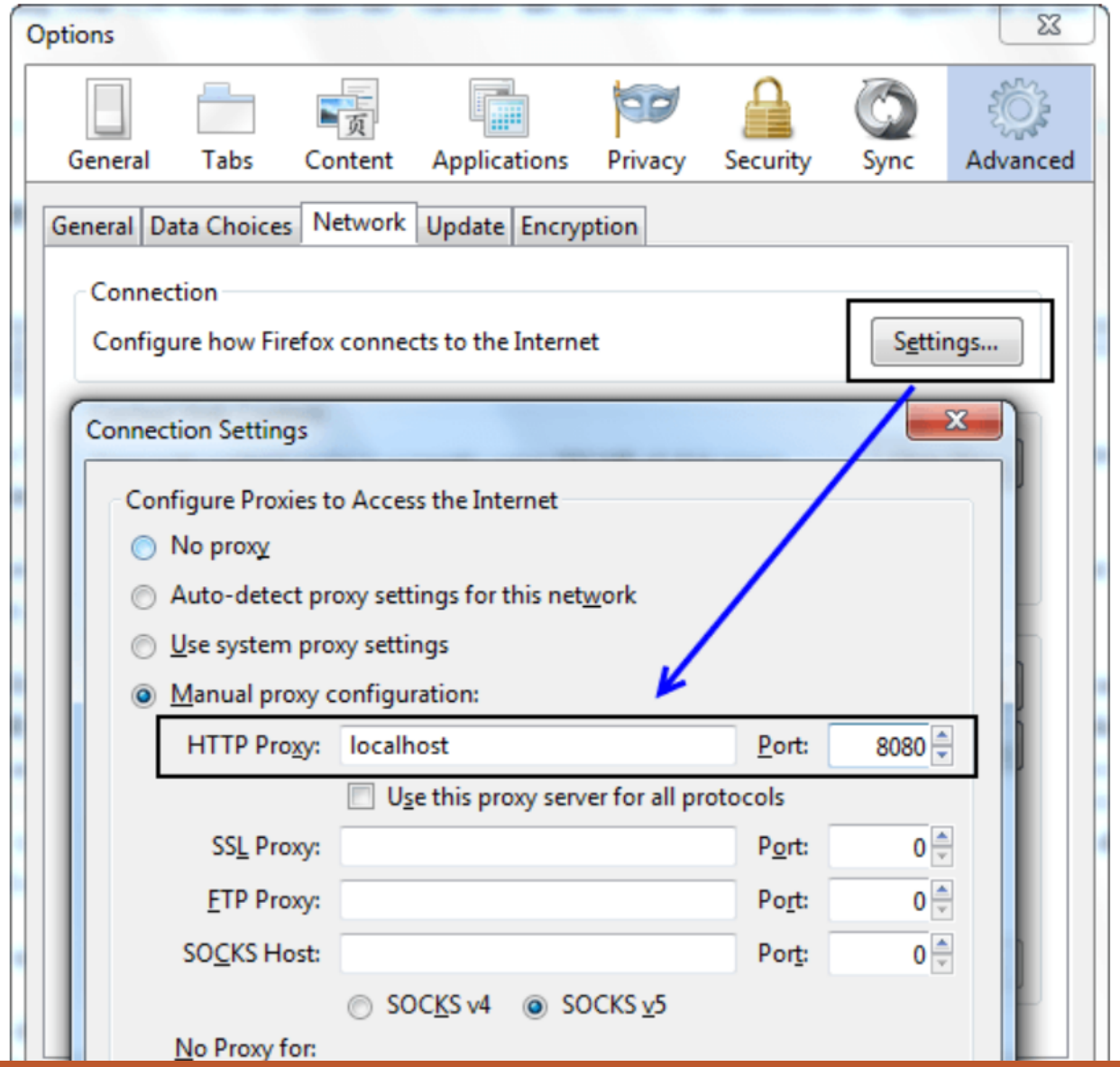


. **Start** Proxy Server

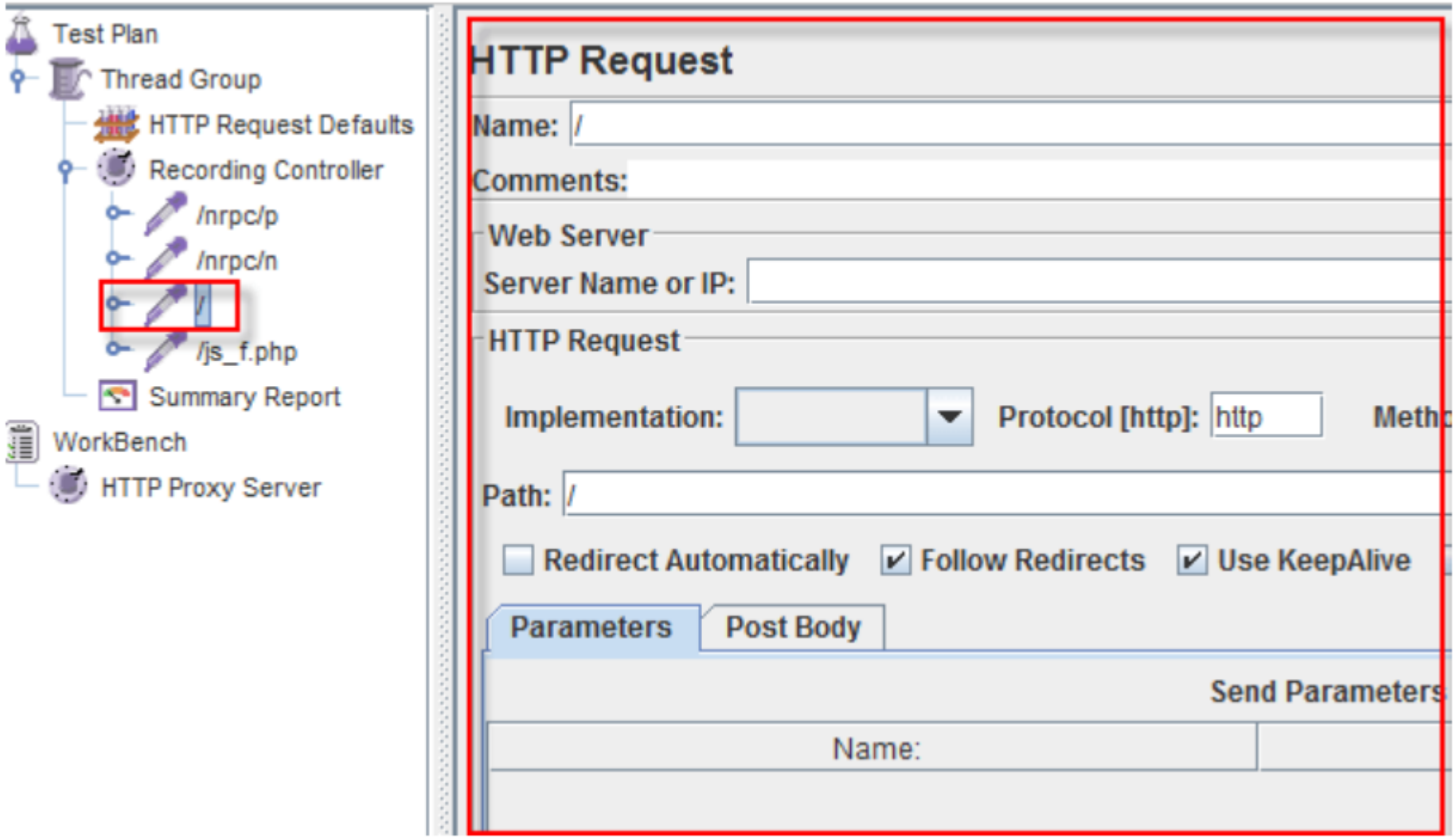
Return to HTTP Proxy Server, and click the **Start** button at the bottom. Now your JMeter proxy server start



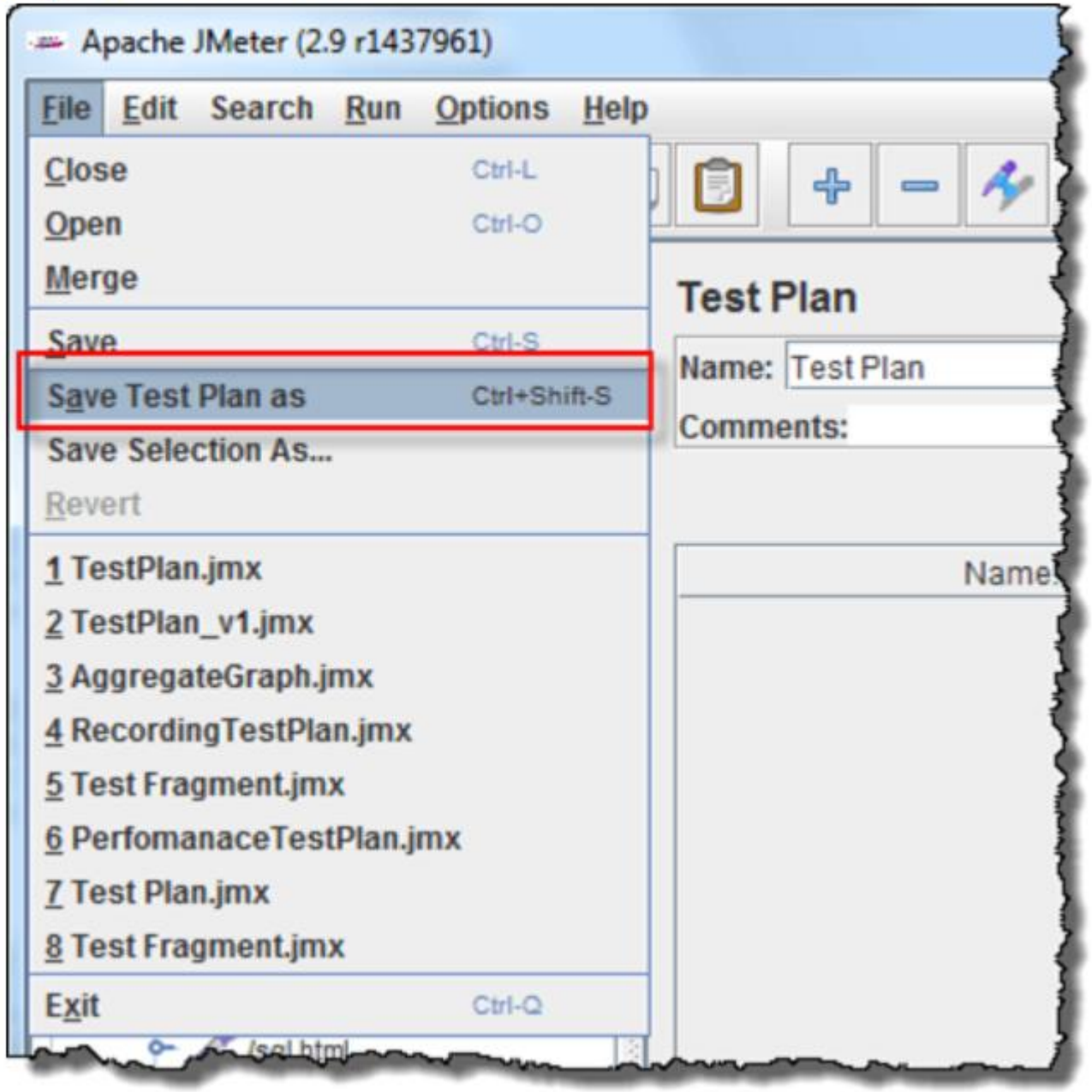
Start your Browser (I used Firefox), choose **Tool** => **Option** => **Advanced** => **Network** => **Setting** => Enter HTTP proxy as figure below



After finishing recording, you will see JMeter automatically created a new HTTP request as the figure below

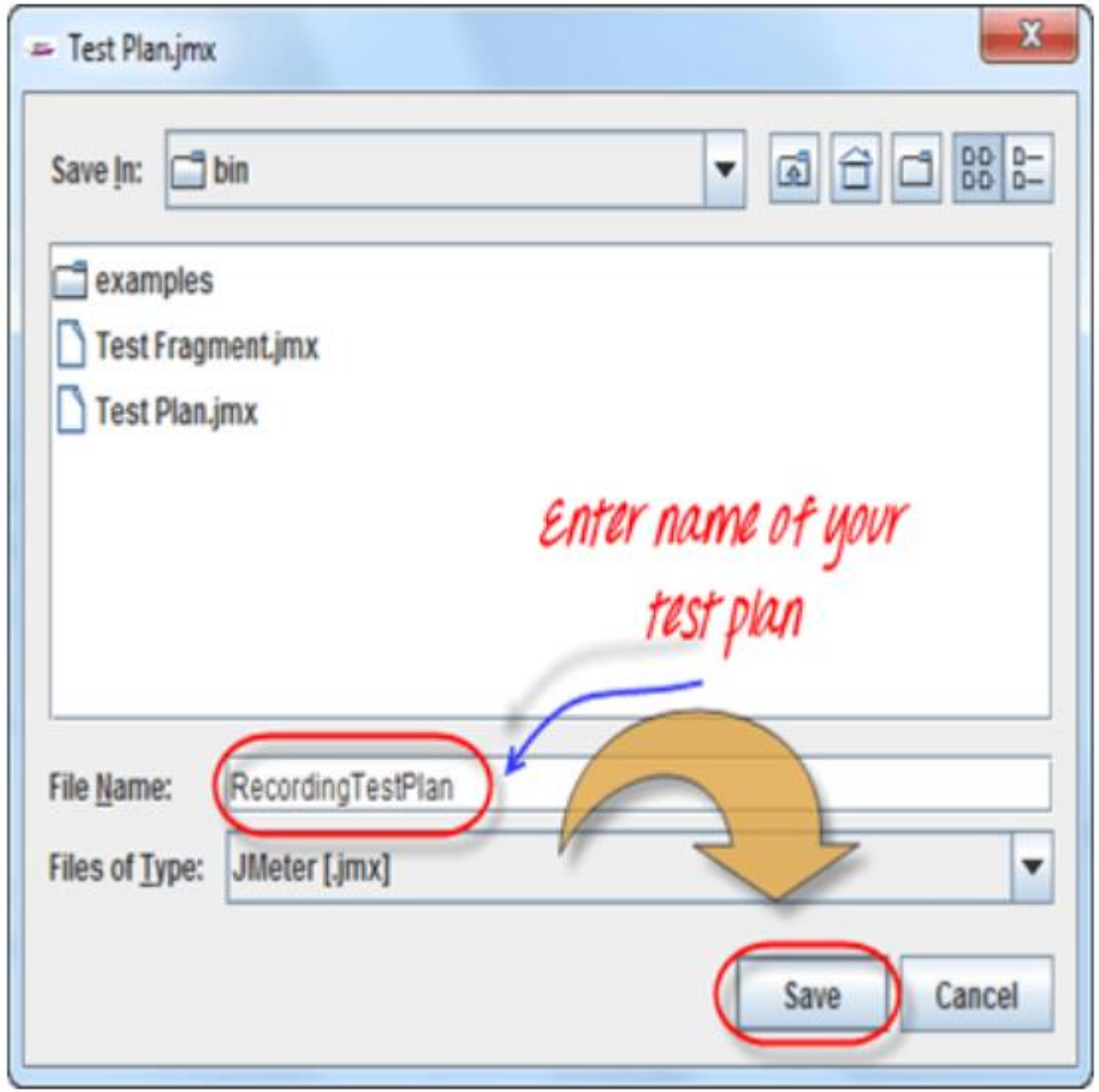


Click File => Save your Test Plan as

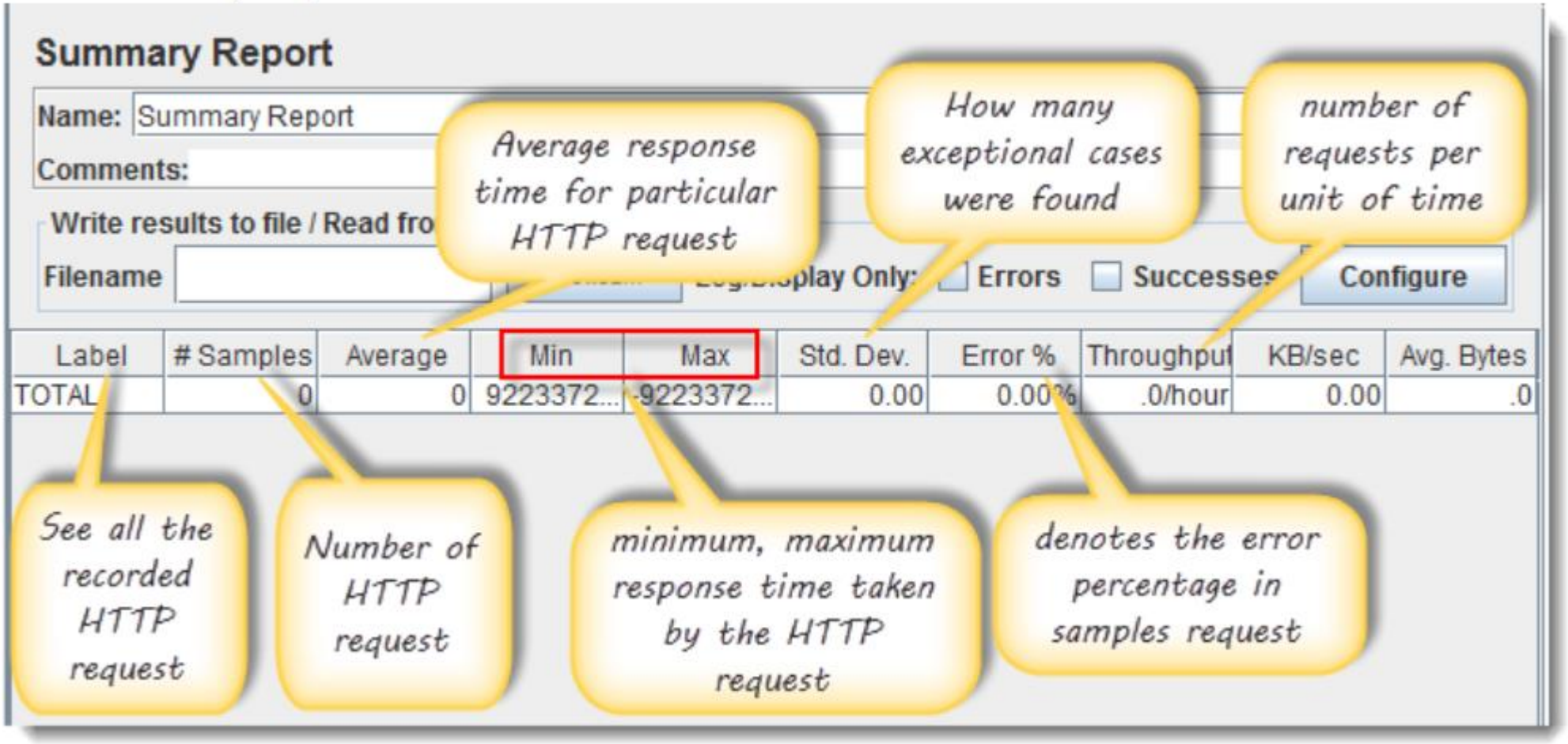


A Dialog box display => enter a name of your test plan at File Name field => Click Save

Now your Test Plan is saved under name RecordingTestPlan.jmx

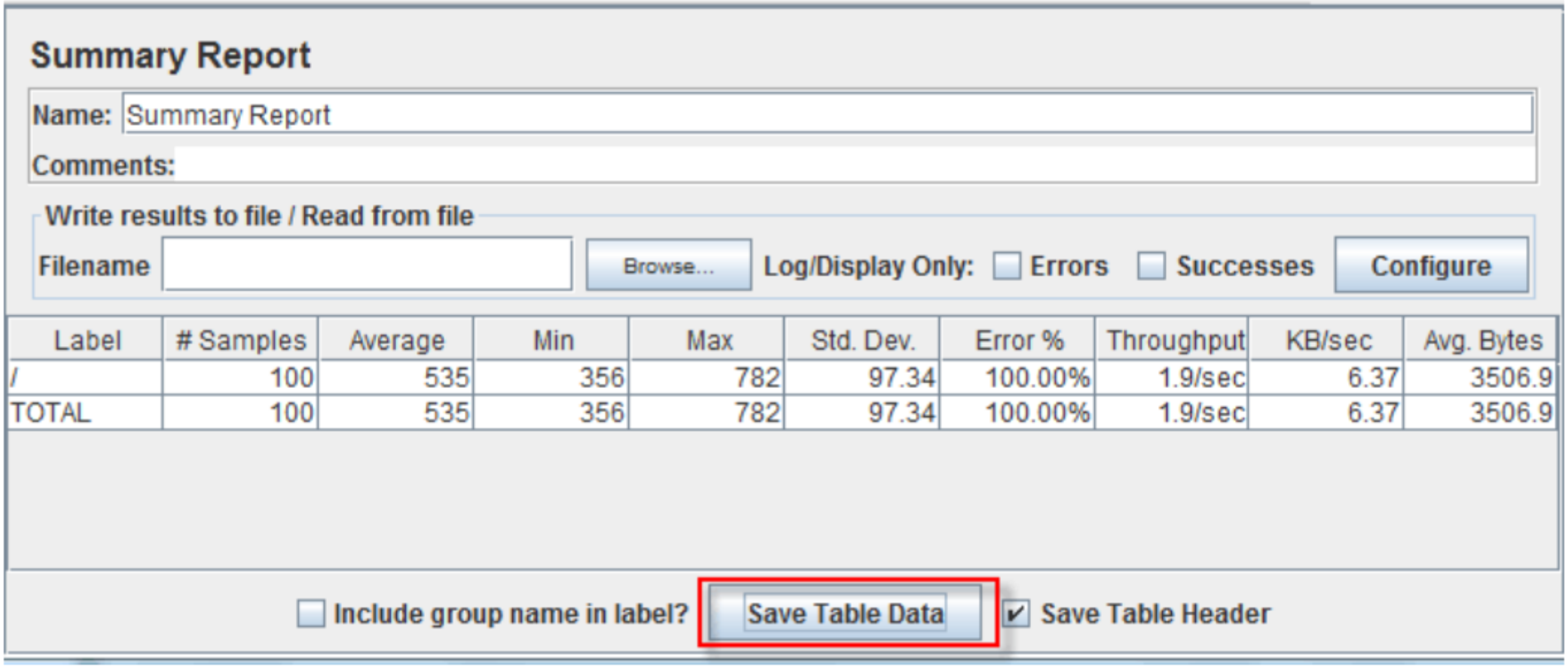


The Summary Report will show some basic statics



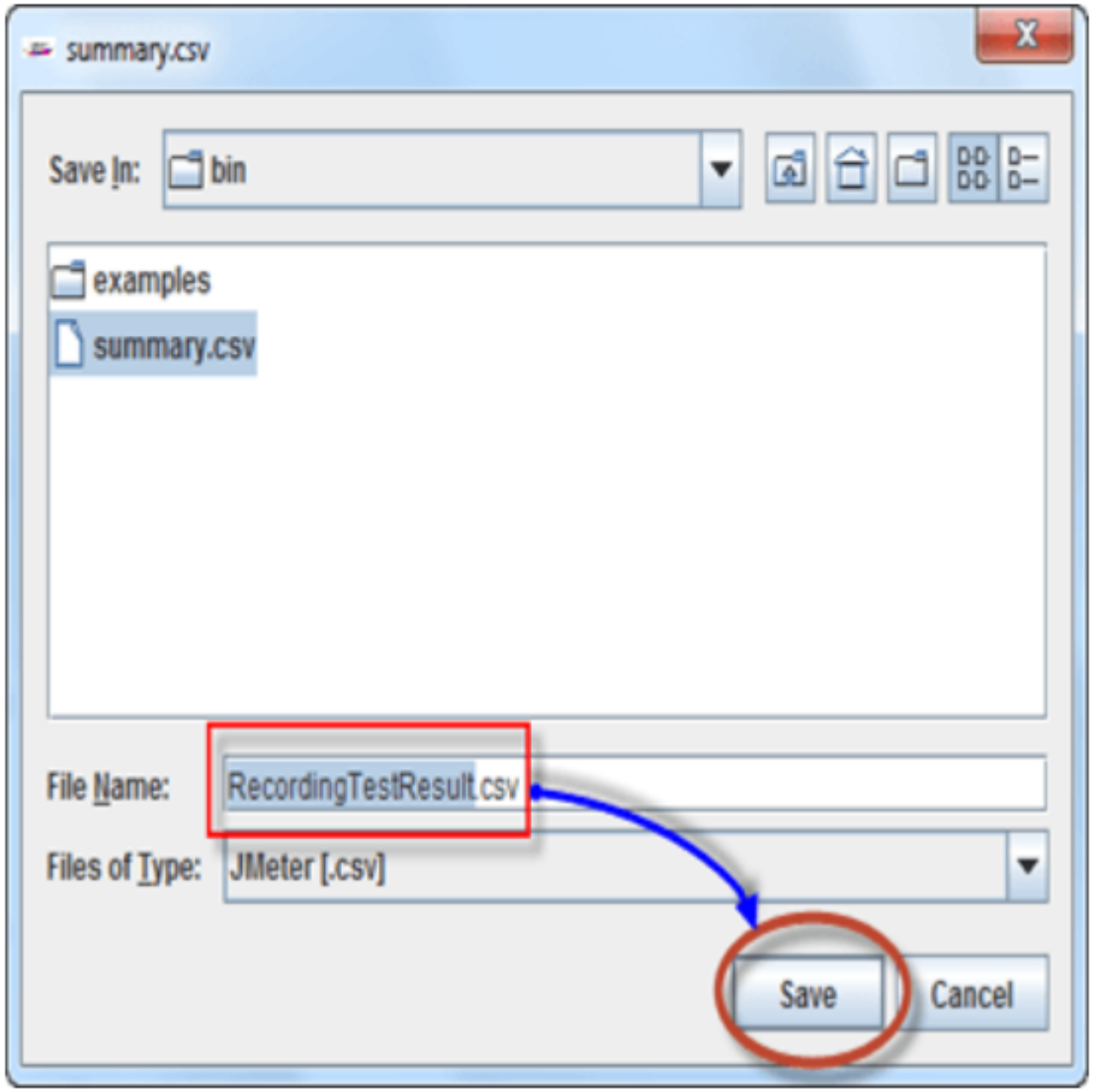
Save your test result

1. Click **Save Table Data** to save test result to file



Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes
/	100	535	356	782	97.34	100.00%	1.9/sec	6.37	3506.9
TOTAL	100	535	356	782	97.34	100.00%	1.9/sec	6.37	3506.9

2. Enter the name of the test result and click Save. Test Result in JMeter is saved in *.csv format as default



Best Practice for JMeter Tests

1) Limit the Number of Threads
2) Using a proxy server
3) Using variables
4) Reduce resource requirement
5) Check the JMeter logs
6) Erase the local path from CSV Data Set Config
7) Follow file naming convention

Best Practice for JMeter Tests

1) **Limit the Number of Threads**
   - The **maximum** number of threads you can effectively run with JMeter is **300**. This limit is because of hardware's capabilities. If JMeter is made to run with more number of threads, the accuracy of timing information will decrease.

2) **Using a proxy server**
   - The Proxy server helps you to abstract out certain common elements from the recorded samples. Moreover, it is useful features to record your testing.

3) **Using variables**
   - Some test plans need to use different values for different users/threads. For example, you might want to test a sequence that requires a unique login for each user. This is easy to achieve using variables.

4) **Reduce resource requirement**
   - The GUI mode consumes a lot of computer memory under heavy load. It causes performance issues

Best Practice for JMeter Tests

Reduce resource requirement

The GUI mode consumes a lot of computer memory under heavy load. It causes performance issues.

There're some tips to reduce resource requirement:

- Use non-GUI mode
- Disable the "View Result Tree" listener during the Load test. Because it consumes more memory and causes JMeter running to run out of memory.
- Disable all JMeter graphs results
- Use the CSV test result format.
- Only save the needed test result. JMeter could take a long time to save very detailed test results.

Best Practice for JMeter Tests

5) Check the JMeter logs

- Any errors in the Test Plan or test execution will be recorded in the log files. Monitoring the log file help you to find the error early

6) Erase the local path from CSV Data Set Config

- If you are using an existing CSV data file that you created on your local computer, you should delete the existing local path (Current path of CSV file). If you don't delete the local path, JMeter cannot find the CSV data file on your local PC.

7) Follow file naming convention

- Don't save test plan under complex file name, use **only alphanumeric** characters

Jmeter Alternatives 2020

SMARTBEAR
LoadNinja

loader.io

Locust

Load runner

StresStimulus
An Easier Way to Load Test™

A distributed Load Testing Tool
TSUNG

LoadView
by Dotcom-Monitor

BlazeMeter

smartmeter.io

Gatling
LOAD TESTING

FunkLoad

References

1. **JMeter Website**
   - https://jmeter.apache.org/
2. **JMeter Tutorial for Beginners: Learn in 7 Days**
   - https://www.guru99.com/jmeter-tutorials.html

THANK YOU