# TABLE OF CONTENT



# Overview

## *Apache JMeter*

- A 100% pure Java application designed to load test client/server software.
- Can be used to test performance both on static and dynamic resources *(Example:- Static files, Java Servlets, ASP.NET, PHP, CGI scripts, Java objects, databases, FTP servers, and more.)*
- Can simulate a heavy load on a server, network or object to test its strength or to analyze overall performance under different load types.

## History

- Started by Stefano Mazzocchi of Apache Software Foundation
- Originally developed to test Apache JServ
- Functionality has been added to test a variety of server-based applications:
  - JDBC
  - LDAP
  - FTP
  - SMTP
  - Etc...

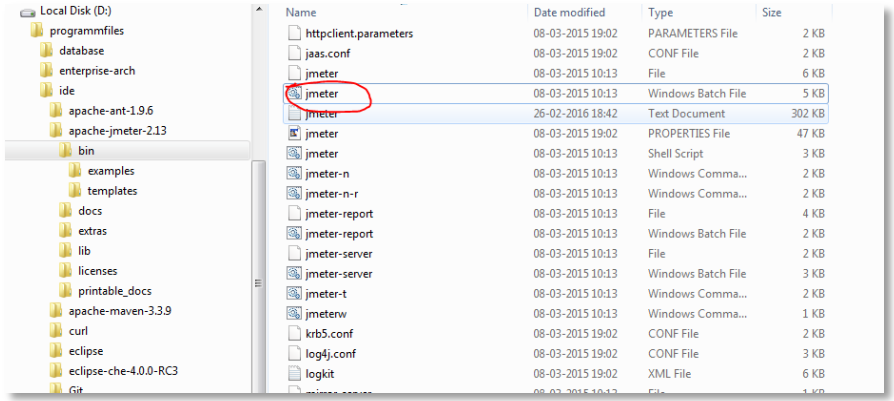
# Installation

## *Pre-requisites*

- Requires Java 6 or Later
- Path defined to %JAVA_HOME%\bin

## Installation

- Download latest JMeter zip (2.13) from below site
  - http://jmeter.apache.org/download_jmeter.cgi
- Extract anywhere of your choice (please make sure java is set in the path)
- Several launch scripts (easiest double click jmeter.bat)




```
apache-jmeter-X.Y
apache-jmeter-X.Y/bin
apache-jmeter-X.Y/docs
apache-jmeter-X.Y/extras
apache-jmeter-X.Y/lib/
apache-jmeter-X.Y/lib/ext
apache-jmeter-X.Y/lib/junit
apache-jmeter-X.Y/licenses
apache-jmeter-X.Y/printable_docs
```

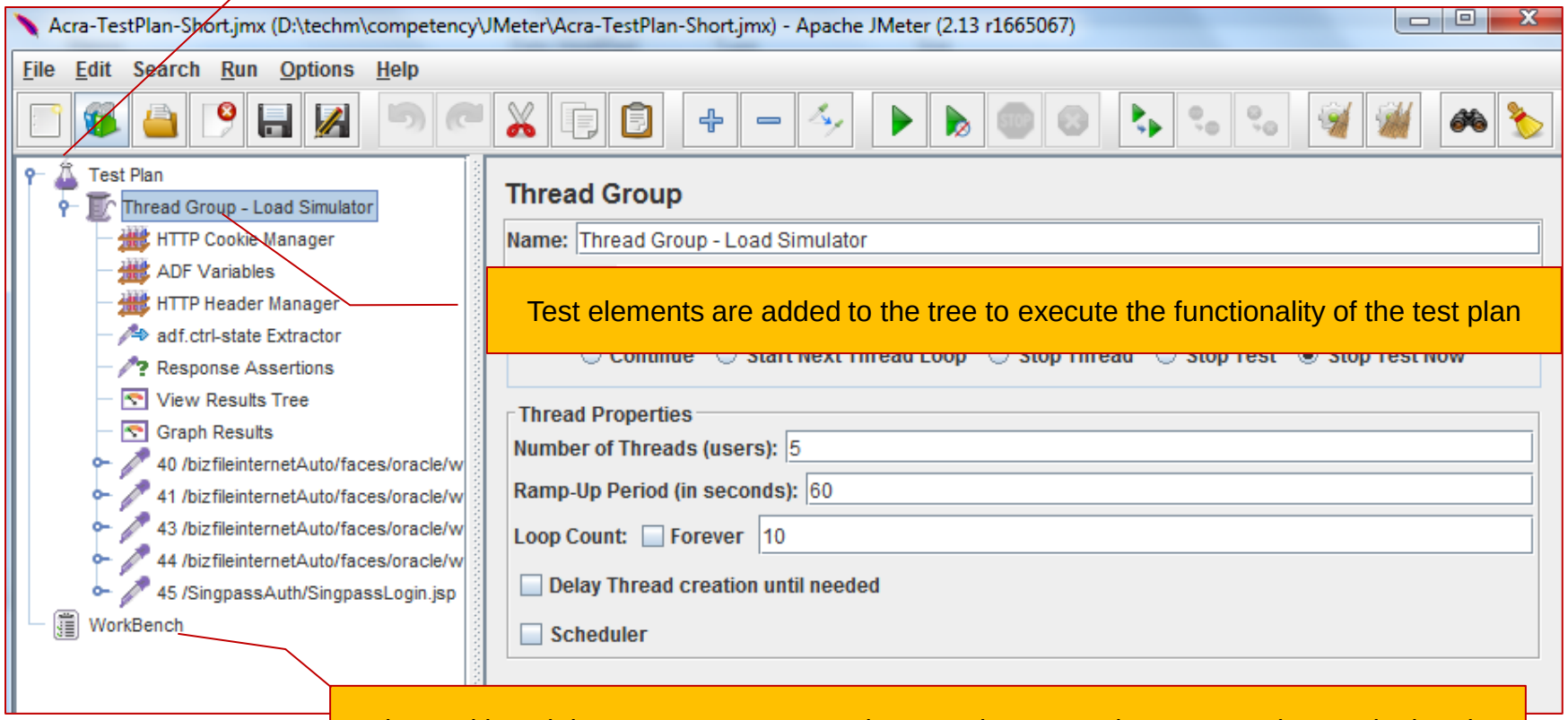
# Starting JMeter

## *Various options to Start*

- jmeter.bat - Standard JMeter in GUI mode
- jmeterw.cmd - JMeter without the windows shell console in GUI mode
- jmeter-n.cmd - JMeter in non-GUI mode (using a JMX file)
- jmeter-n-r.cmd - JMeter remotely in non-GUI mode for  (using a JMX file)
- jmeter-t.cmd - JMeter in GUI mode (using a JMX file)
- jmeter-server.bat - Start JMeter in server mode
- mirror-server.cmd - Runs the JMeter Mirror Server in non-GUI mode
- shutdown.cmd - Run the Shutdown client to stop a non-GUI instance gracefully
- stoptest.cmd - Run the Shutdown client to stop a non-GUI instance abruptly

# Focus Areas

Test Plan is the parent element, describes series of steps to run

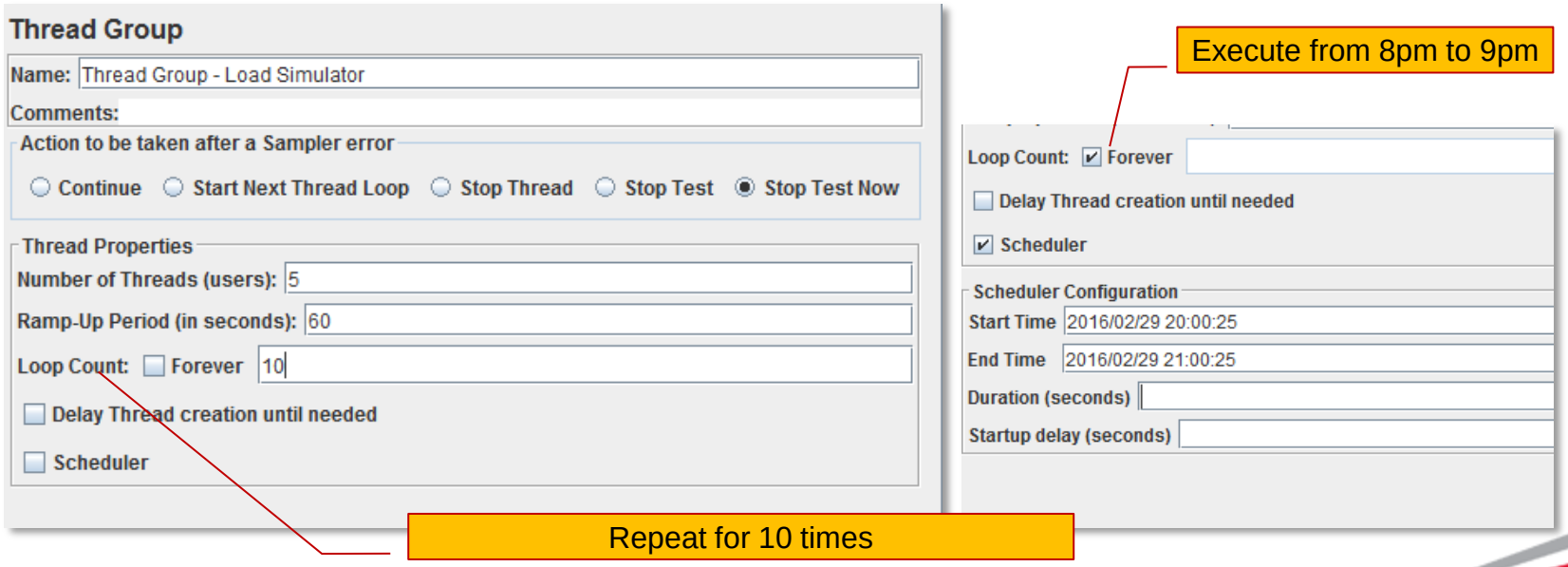



Test elements are added to the tree to execute the functionality of the test plan

The workbench is a temporary parent element where test elements can be manipulated.

# Test Elements : Thread Group

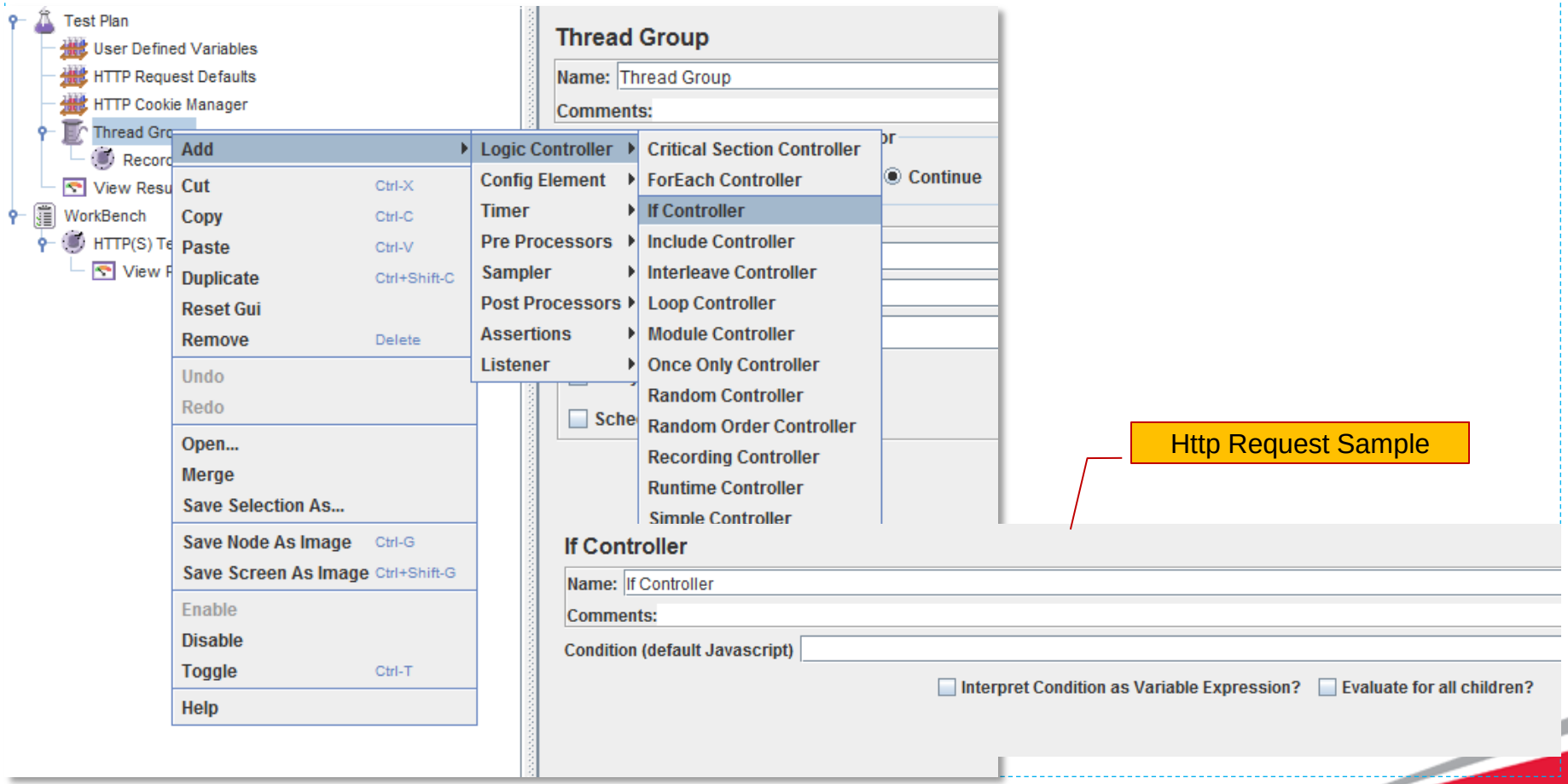
- Number of Threads : Number of virtual users.
- Thread Group : This is the place where number of threads, ramp-up period, loop count are given while executing.
- Ramp-Up Period : It indicates the time taken by J-Meter to create all of the threads needed. If we set 10 seconds as the ramp-up period for 5 threads then the J-Meter will take 10 seconds to create those 5 threads. Also by setting its value to 0, all the threads can be created at once.
- Loop Count : By specifying its value J-meter gets to know that how many times a test is to be repeated.



# Test Elements : Logic Controllers

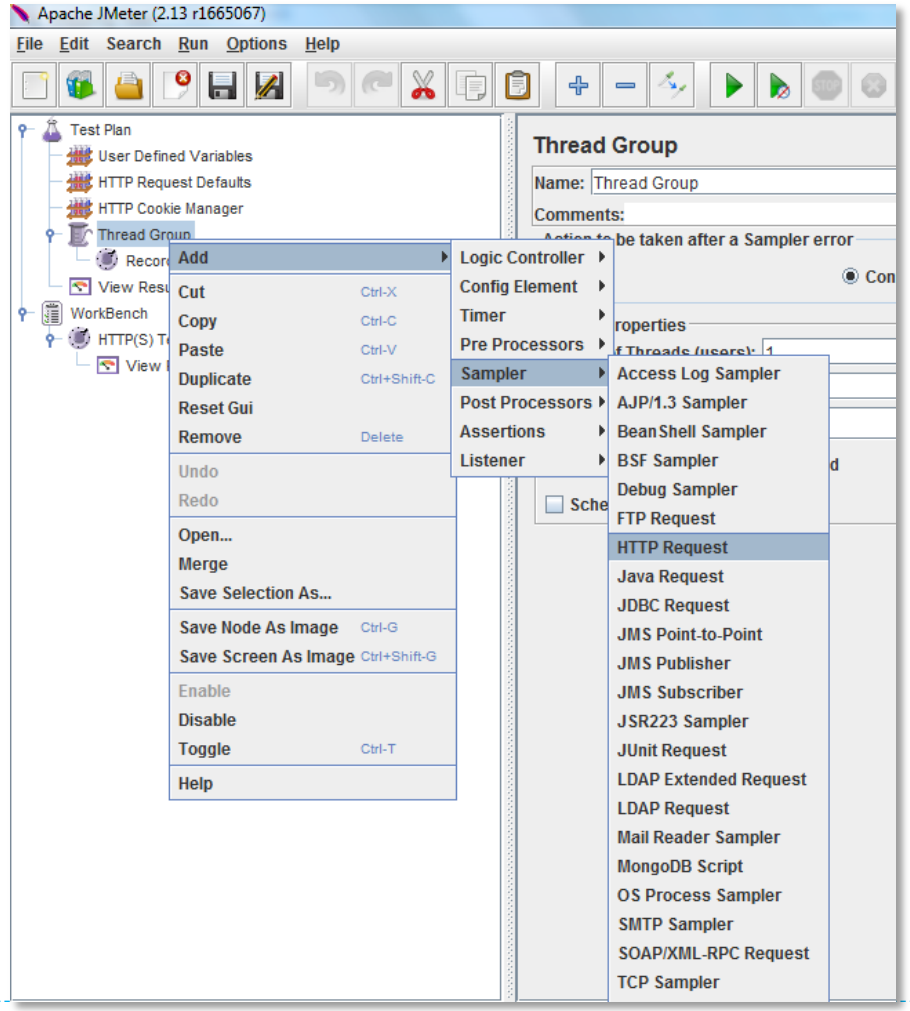
- JMeter has two types of Controllers:
  - Samplers and Logical Controllers.
  - These drive the processing of a test.

- Logic Controllers : Logic Controllers helps to customize the logic of when to send requests.

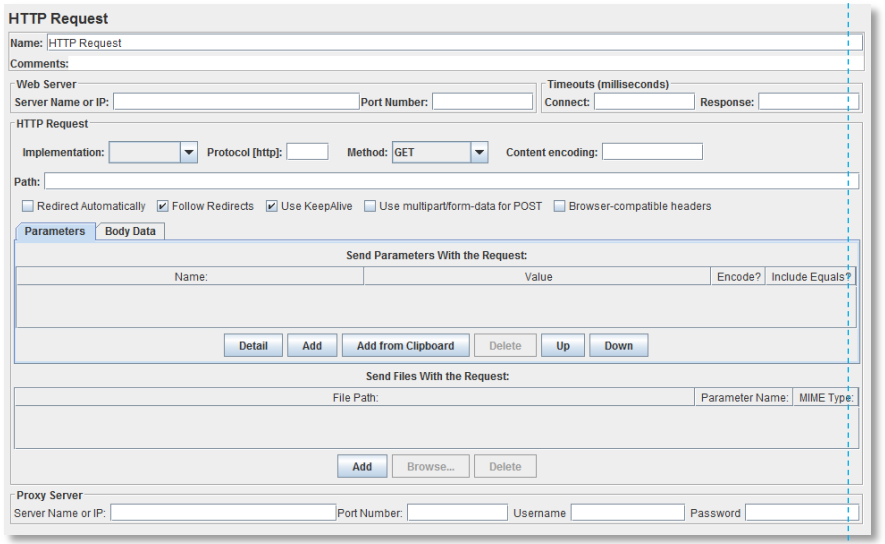


# Test Elements : Samplers

- Samplers : Samplers tell JMeter to send requests to a server and wait for a response.
- They are processed in the order they appear in the tree.

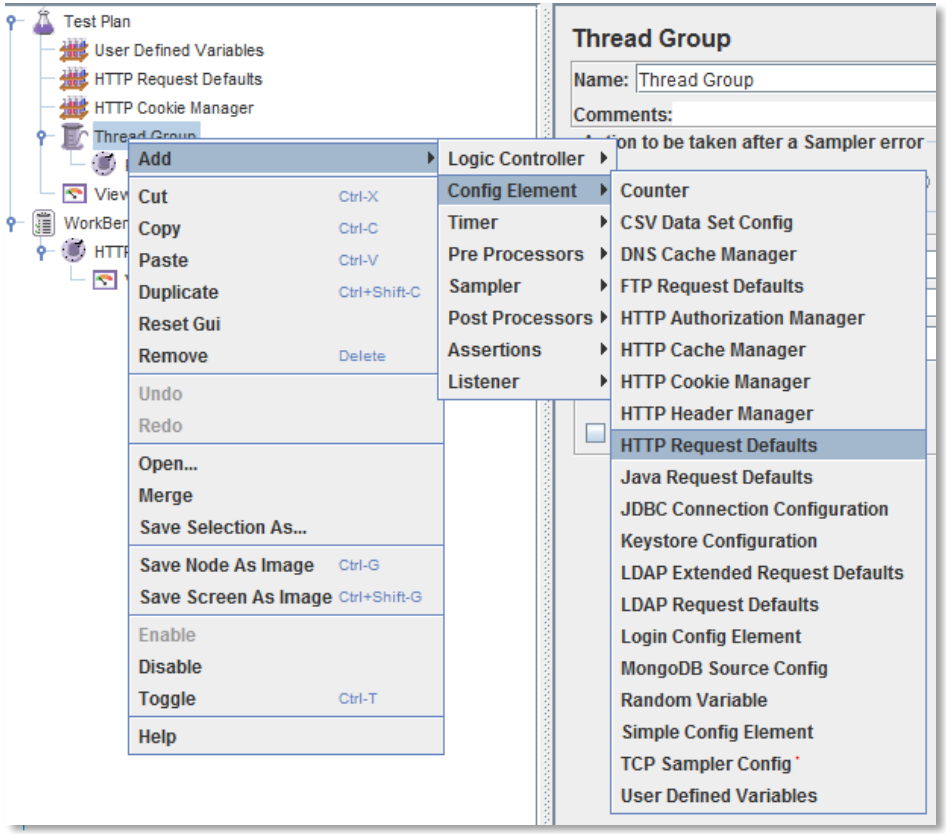


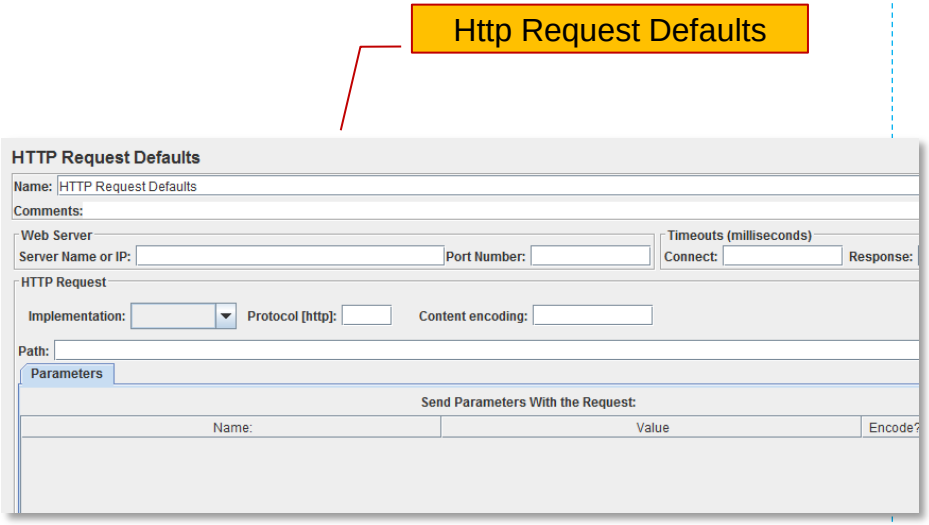

Http Request Sample



# Test Elements : Config Elements

- Config Element: A configuration element works closely with a Sampler.
- It does not send requests (except for HTTP(S) Test Script Recorder), it can add to or modify requests.
- A configuration element is accessible from only inside the tree branch where it is placed the element

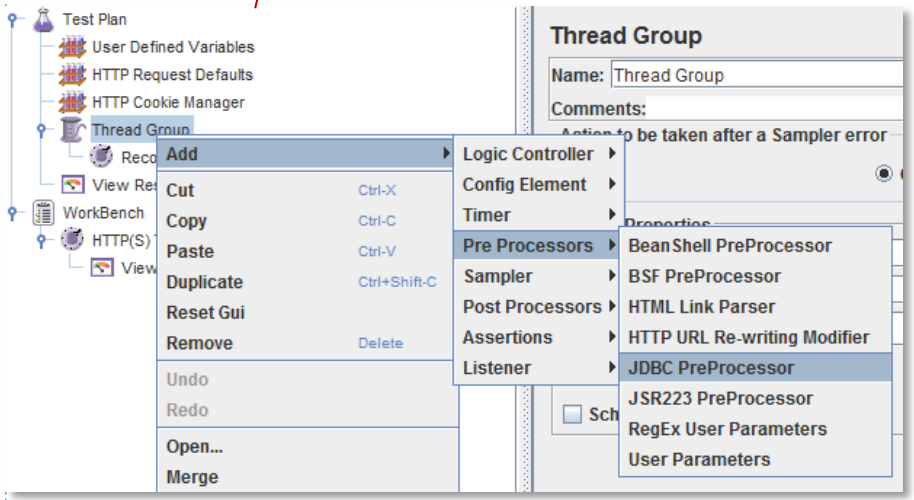


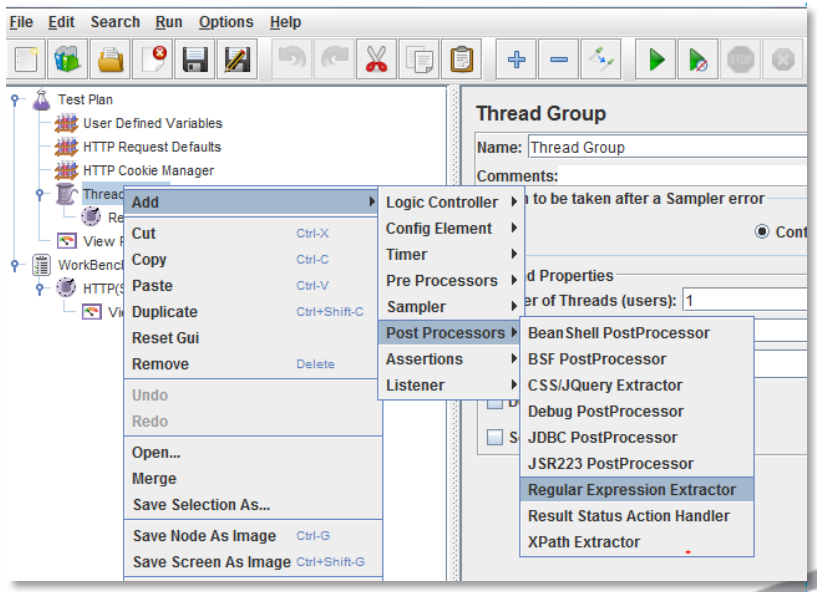
# Test Elements : Pre and Post Processors

- Pre-processor :
  - Pre-Processor executes some action prior to a Sampler Request being made.
  - Pre-Processor is most often used to modify the settings of a Sample Request just before it runs, or to update variables that aren't extracted from response text..
- Post-processor :
  - Post-Processor executes some action after a Sampler Request has been made.
  - Post-Processor is most often used to process the response data, often to extract values from it.

Pre-Processor



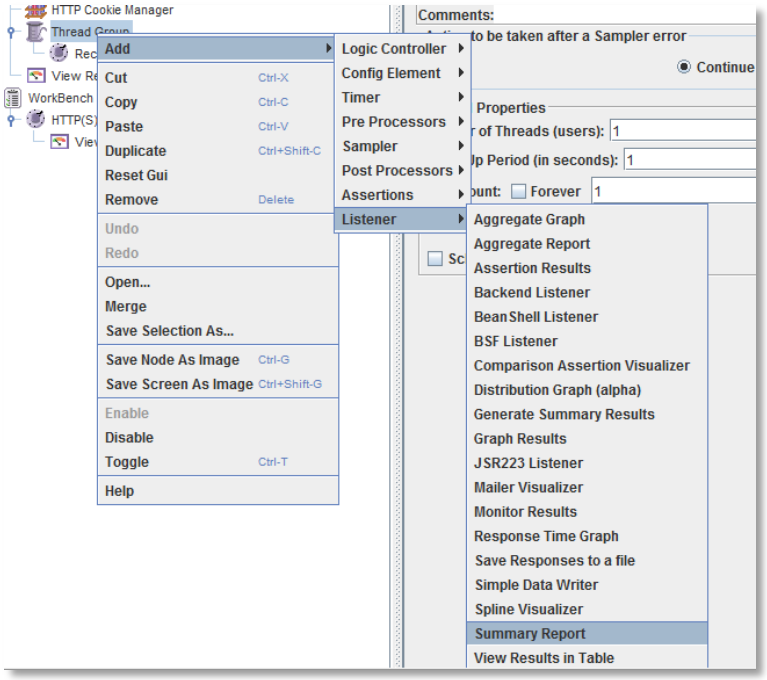

Post-Processor



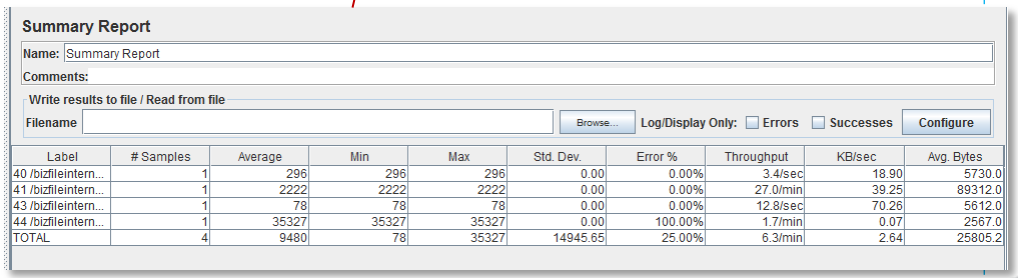
# Test Elements : Listeners

- Listeners : Listeners provide access to the information JMeter gathers about the test cases while JMeter runs.
  - The Graph Results listener plots the response times on a graph.
  - The "View Results Tree" Listener shows details of sampler requests and responses, and can display basic HTML and XML representations of the response.
  - Other listeners provide summary or aggregation information.

- Listeners can direct the data to a file for later use, and configuration button helps to choose which fields to save, and whether to use CSV or XML format



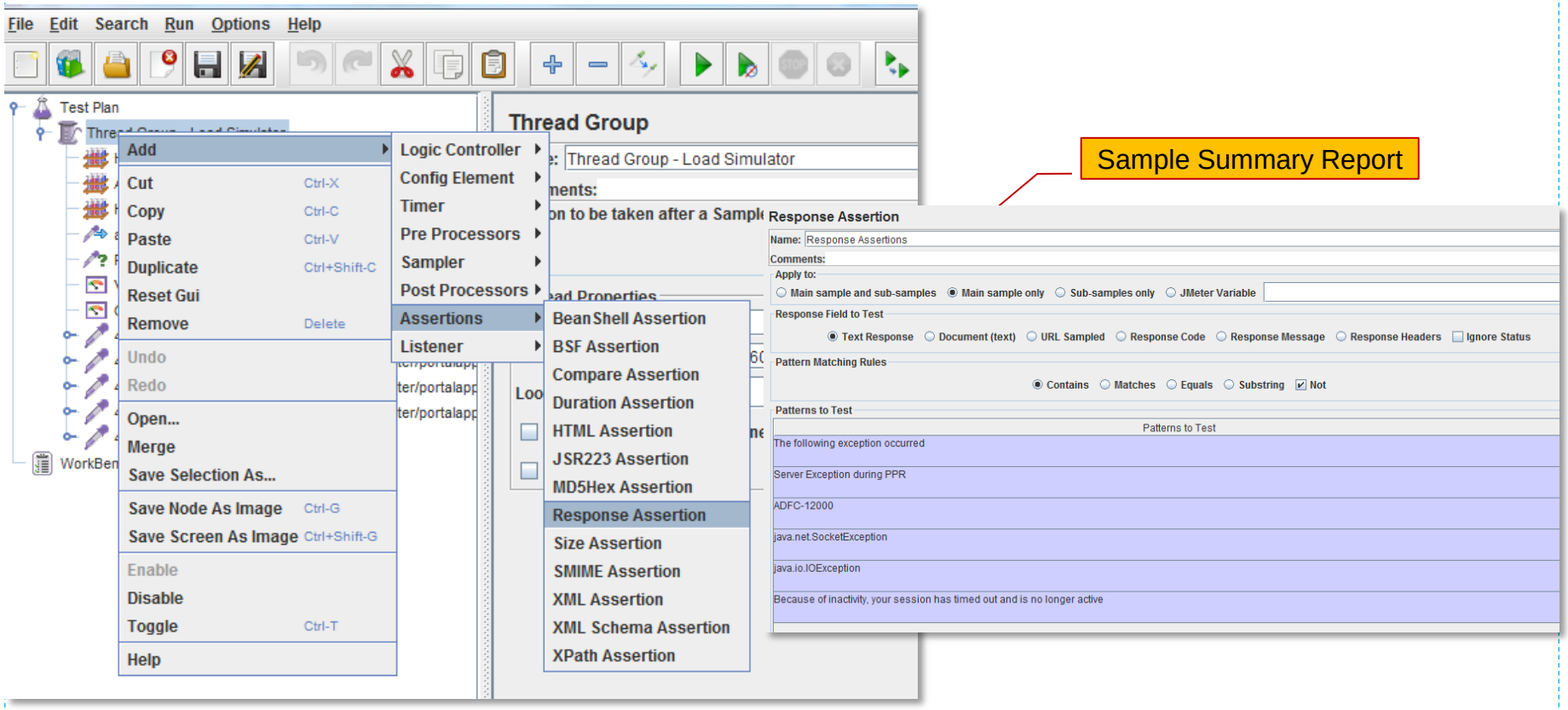

Sample Summary Report




| Label | # Samples | Average | Min | Max | Std. Dev. | Error % | Throughput | KB/sec | Avg. Bytes |
|---|---|---|---|---|---|---|---|---|---|
| 40 /bizfileintern... | 1 | 296 | 296 | 296 | 0.00 | 0.00% | 3.4/sec | 18.90 | 5730.0 |
| 41 /bizfileintern... | 1 | 2222 | 2222 | 2222 | 0.00 | 0.00% | 27.0/min | 39.25 | 89312.0 |
| 43 /bizfileintern... | 1 | 78 | 78 | 78 | 0.00 | 0.00% | 12.8/sec | 70.26 | 5612.0 |
| 44 /bizfileintern... | 1 | 35327 | 35327 | 35327 | 0.00 | 100.00% | 1.7/min | 0.07 | 2567.0 |
| TOTAL | 4 | 9480 | 78 | 35327 | 14945.65 | 25.00% | 6.3/min | 2.64 | 25805.2 |

# Test Elements : Assertions

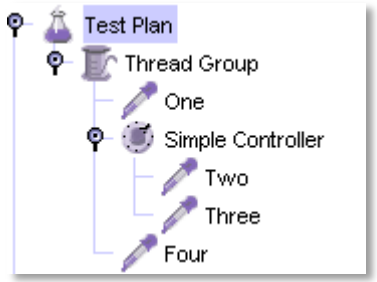
- Assertions:
  - Allow assert facts about responses received from the server being tested.
  - We can "test" that application is returning the results you expect it to.
  - *Example : We can assert that the response to a query will contain some particular text.*

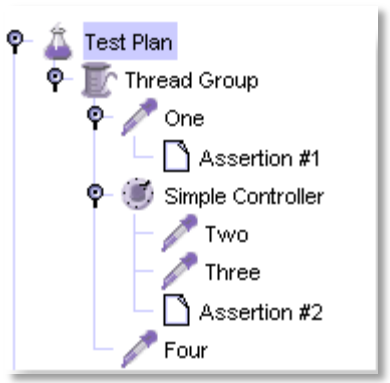


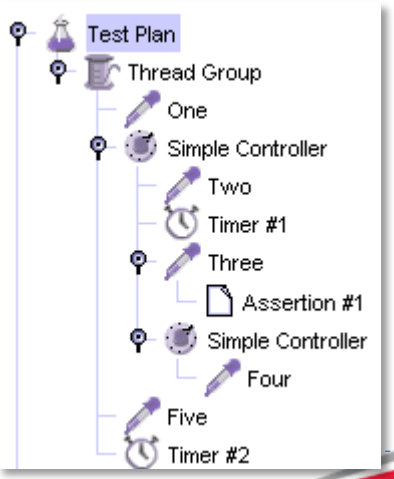
# Execution Order & Scoping Rules

- Execution Order:
  1. Configuration elements
  2. Pre-Processors
  3. Timers
  4. Sampler
  5. Post-Processors
  6. Assertions
  7. Listeners

- Scoping Rules:
  - The JMeter test tree contains elements that are both hierarchical and ordered.
    - Some elements in the test trees are strictly hierarchical (Listeners, Config Elements, Post-Procesors, Pre-Processors, Assertions, Timers), and
    - Some are primarily ordered (controllers, samplers).

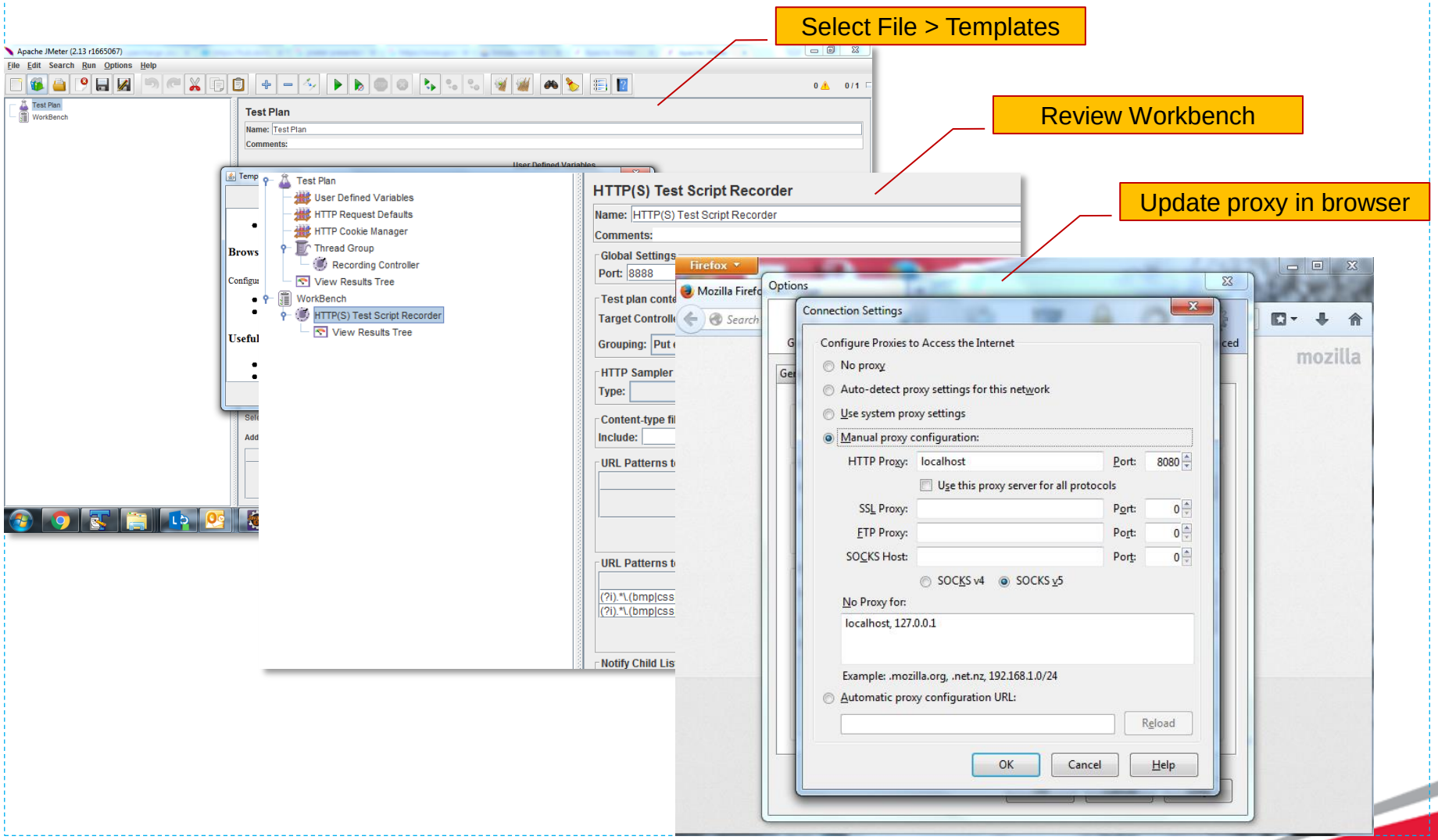


# Recording JMeter – Test Plan

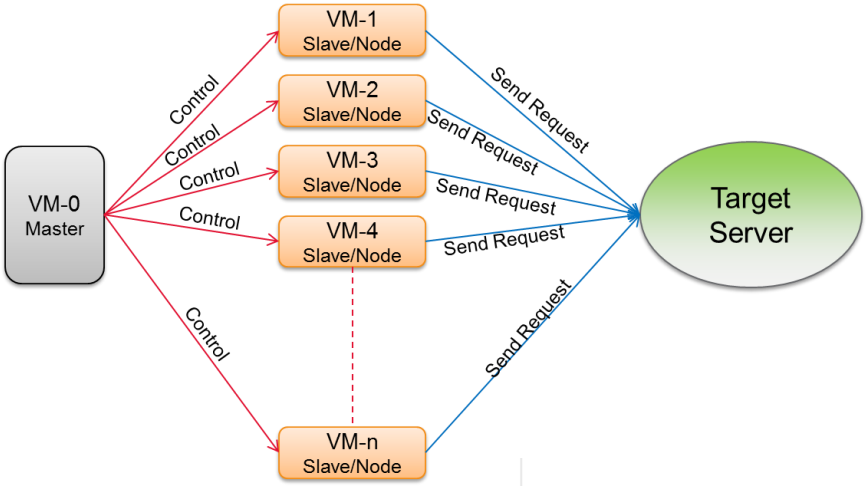
- Test Plans can be created from existing template.
- Use the menu File > Templates... or Templates icon



# Scaling JMeter

- **Distributed Testing : To distribute the load of JMeter across multiple slaves/nodes, run JMeter in server mode on the remote node(s), and then control the server(s) from the JMeter Master GUI.**

- Code base for master and slave JMeter are same, except how they are started and the configurations defined




IP addrres of each slave

```
# Remote Hosts - comma delimited
remote_hosts=192.168.0.10,192.168.0.11,192.168.0.12
#remote_hosts=localhost:1099,localhost:2010
```

- Steps for distributed testing :
  1. On Slave : Go to JMeter/bin and execute jmeter-server.sh
  2. On master systems: Go to /bin directory and edit file jmeter.properites, add IP of slave machine
  3. On the master machine, run JMeter GUI and open the test plan
  4. Click Run on the menu bar; select Remote start -> select the IP address of slave machine

# Best Practices

- Always use latest version of JMeter
- Use the correct Number of Threads
- Where to Put the Cookie Manager
- Where to Put the Authorization Manager
- Using the HTTP(S) Test Script Recorder
- User variables
- Reducing resource requirements
- BeanShell server

Apache JMeter™

# *Demo*

# Thank you

## Visit us at www.techmahindra.com

**Disclaimer**

Tech Mahindra Limited, herein referred to as TechM provide a wide array of presentations and reports, with the contributions of various professionals. These presentations and reports are for informational purposes and private circulation only and do not constitute an offer to buy or sell any securities mentioned therein. They do not purport to be a complete description of the markets conditions or developments referred to in the material. While utmost care has been taken in preparing the above, we claim no responsibility for their accuracy. We shall not be liable for any direct or indirect losses arising from the use thereof and the viewers are requested to use the information contained herein at their own risk. These presentations and reports should not be reproduced, re-circulated, published in any media, website or otherwise, in any form or manner, in part or as a whole, without the express consent in writing of TechM or its subsidiaries. Any unauthorized use, disclosure or public dissemination of information contained herein is prohibited. Unless specifically noted, TechM is not responsible for the content of these presentations and/or the opinions of the presenters. Individual situations and local practices and standards may vary, so viewers and others utilizing information contained within a presentation are free to adopt differing standards and approaches as they see fit. You may not repackage or sell the presentation. Products and names mentioned in materials or presentations are the property of their respective owners and the mention of them does not constitute an endorsement by TechM. Information contained in a presentation hosted or promoted by TechM is provided “as is” without warranty of any kind, either expressed or implied, including any warranty of merchantability or fitness for a particular purpose. TechM assumes no liability or responsibility for the contents of a presentation or the opinions expressed by the presenters. All expressions of opinion are subject to change without notice.