What is JMeter

- Tool or Application for performance testing
- Free & Open Source
- Built with Java

Features of JMeter :

- Completely free
- for performance testing of a variety of applications
- Web | API | FTP | DB
- Option for Recording
- Inbuilt components for adding assertions & reporting
- Extensible using plugins
- JMeter is a server-side performance testing tool

How to setup JMeter:

1 - Check java is installed java -version-15 LINK
2 - Download JMeter LINK
3 - Unzip
4 - Start JMeter
- bin/jmeter.bat (win)
- bin/jmeter.sh (mac/linux)

How to prepare for a Performance Test:

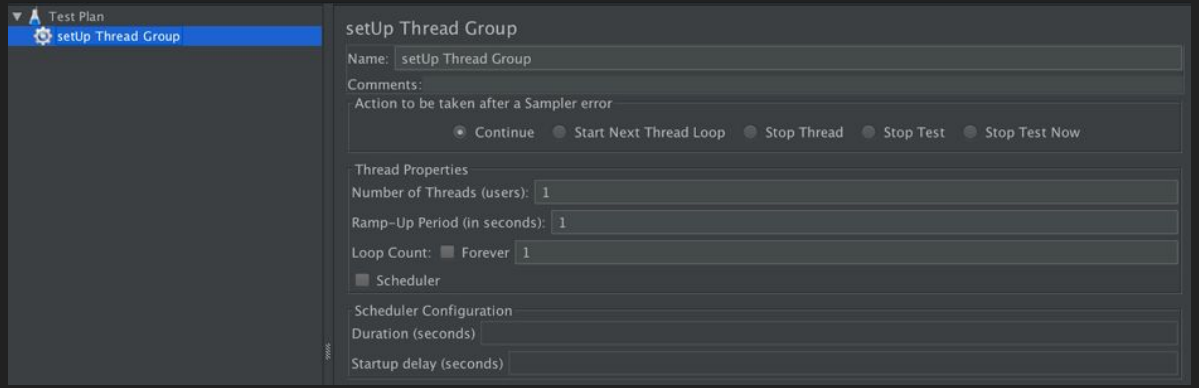
- Get approvals & authorizations
- Application & Environment
- Requirements for the test - What to test, Scenarios
- Test Data
- Expected behavior or performance OR Baseline
- Reporting Format

Test Plan:

- is a like a project where all the elements of the performance test are added
- It saves as a .jmx file
- Inside JMeter the Test Plan stores & displays the elements of the test in a tree view
- A minimal test will consist of the Test Plan, a Thread Group & one or more Samplers LINK

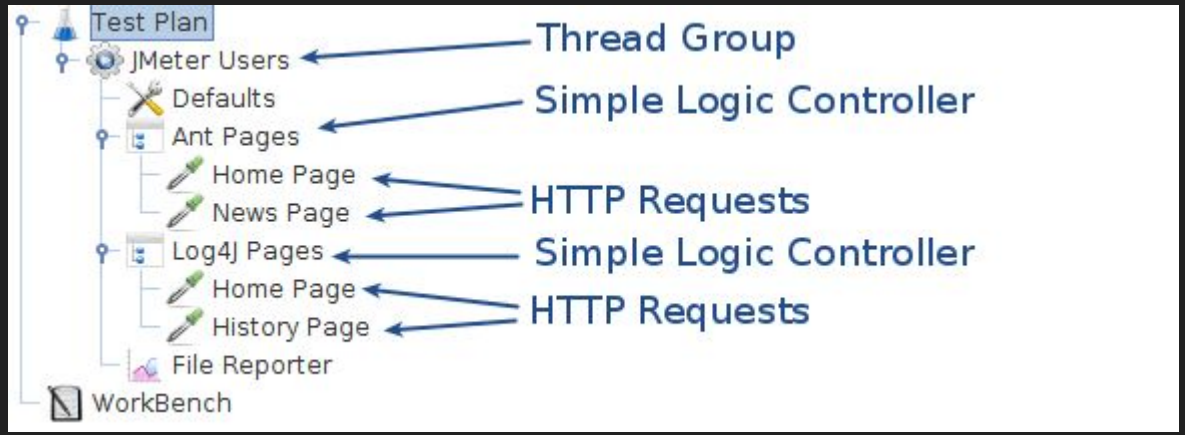
Thread Group:

- A Thread Group is a collection of threads
- Every JMeter test plan starts with this element
- Each thread represents one user using the application under test
- In a thread group element, we can define the
- No of the users (threads)
- Ramp up & down
- Loops / iterations



Controllers:

JMeter has two types of Controllers - Samplers & Logical Controllers



Samplers:

- Tell JMeter to send requests to a server & wait for a response
- They are processed in the order they appear in the tree
- Each sampler (except Flow Control Action) generates one or more sample results

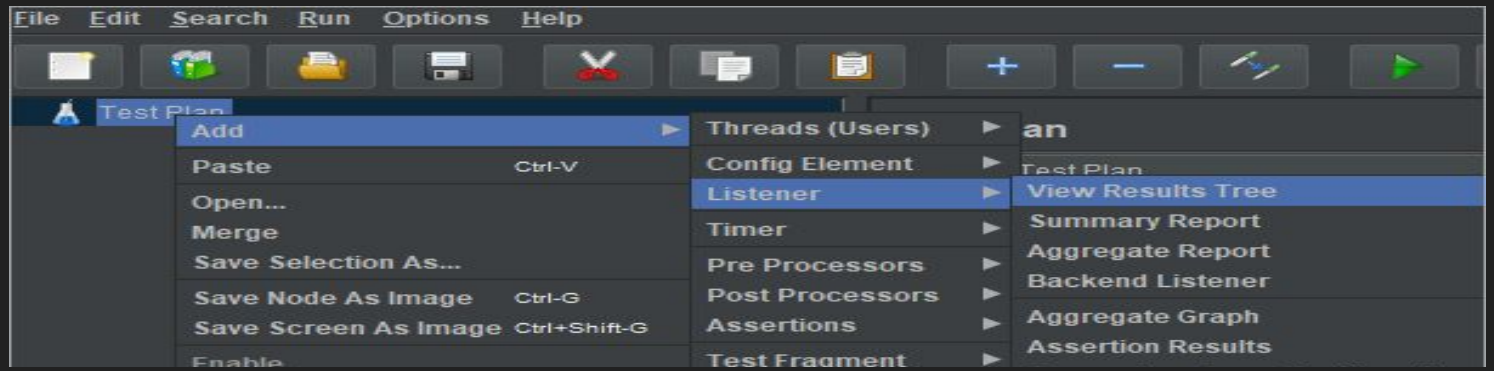
Logical Controllers:

determine the order in which Samplers are processed

let you customize the logic that JMeter uses to decide when to send requests

Listeners:

- Components that show the result of the samplers
- Tree | Table | Graphs
- also provide means to view, save, read saved test results
- Listeners are processed at the end of the scope in which they are found
- Listeners can use a lot of memory if there are a lot of samples
- To minimize the amount of memory needed, use the Simple Data Writer

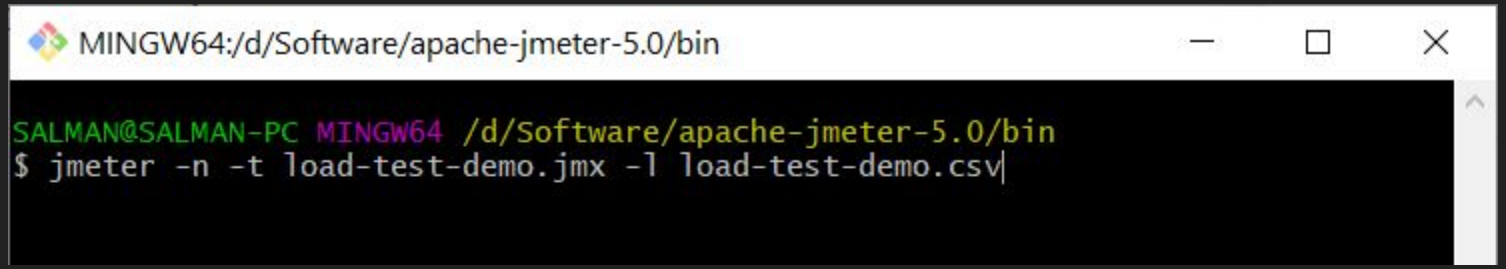


How to generate Report:

On cmd prompt

Goto the location of JMeter bin folder

jmeter -n -t “location of imx file” -l “location of result file csv”

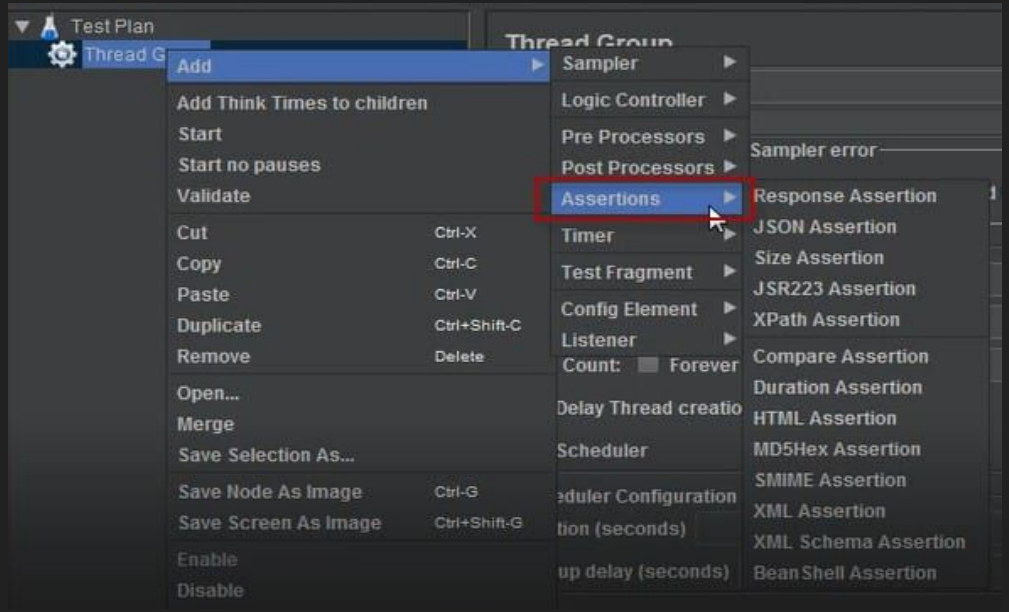


Assertions:

Assertion = Check or Validation

Check Actual = Expected

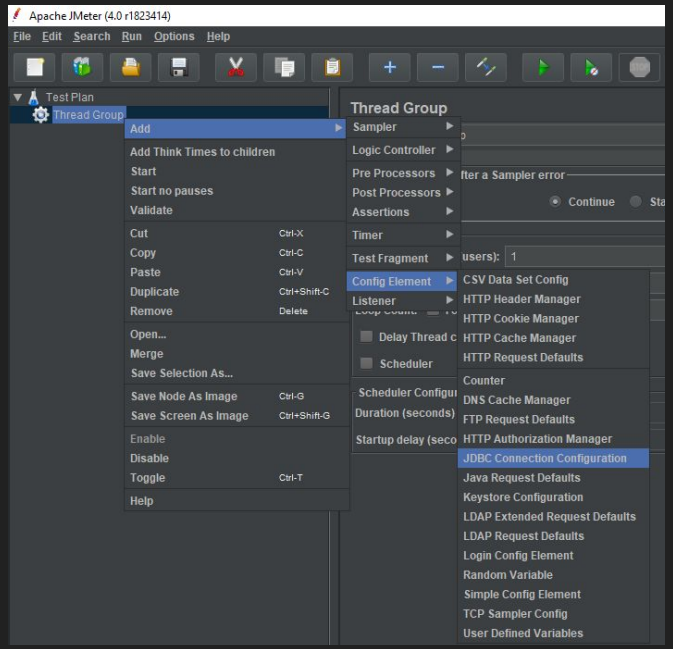
processed after every sampler in the same scope



Config Elements:

to set up defaults & variables for later use by samplers

processed at the start of the scope in which they are found, i.e. before any samplers in the same scope



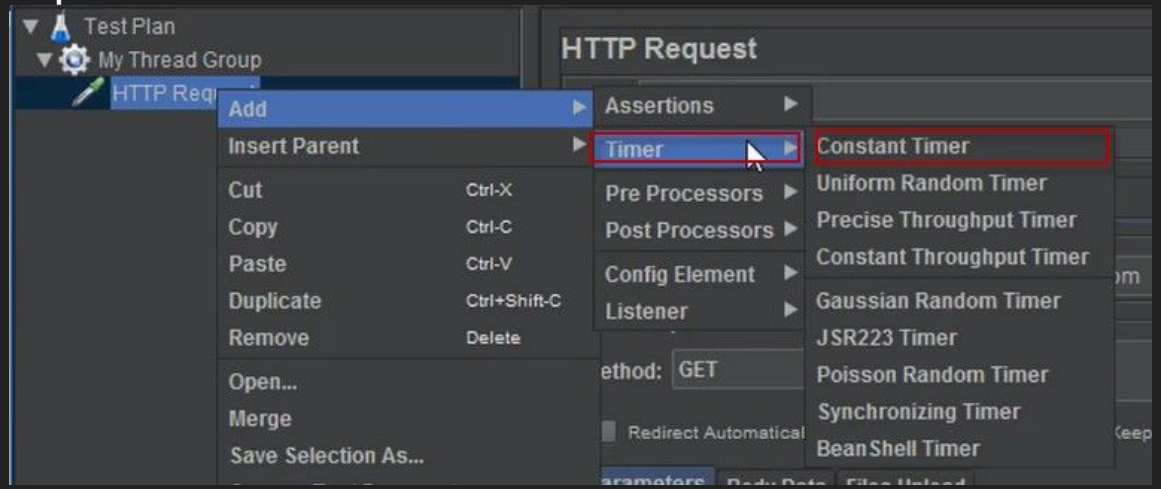
Timers:

to add delay (pause) before sampler request

By default, JMeter thread executes samplers in sequence without pausing

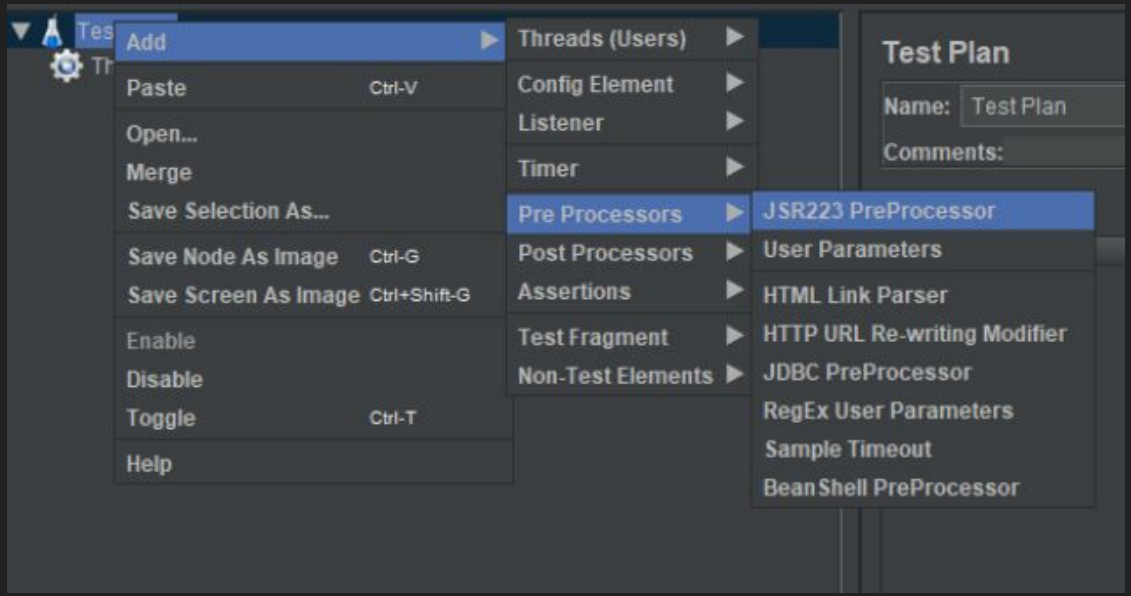
If more than 1 timers are used in Thread Group, JMeter will take the sum of the timers to pause

can be added as children of samplers or controllers



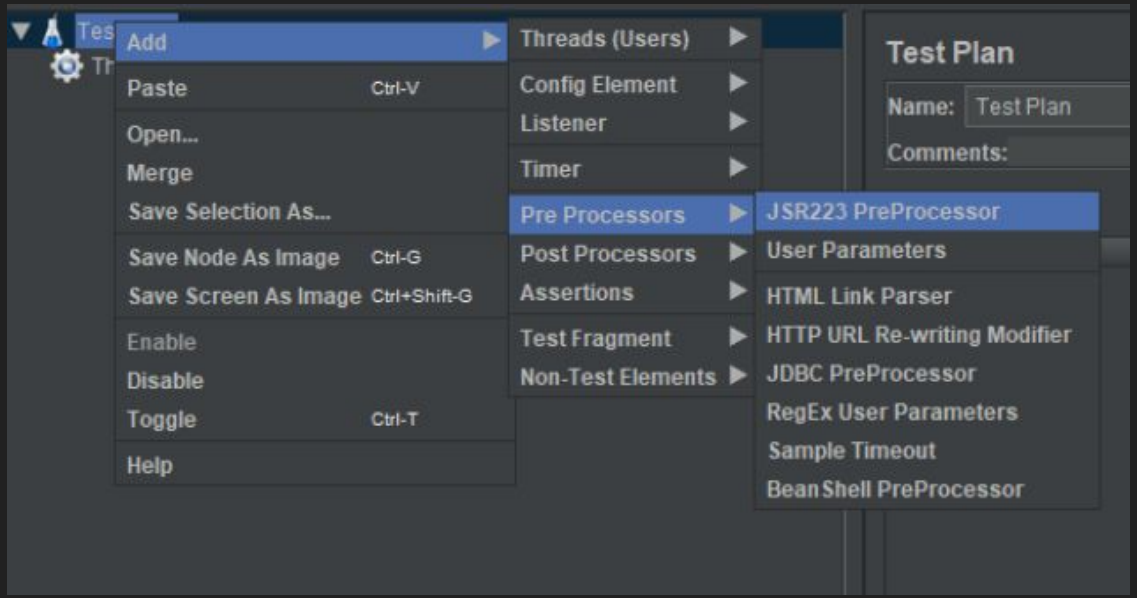
Pre-Processors:

executes prior to a Sampler Request
modify the settings of a Sample Request Or update variables



Post-Processors:

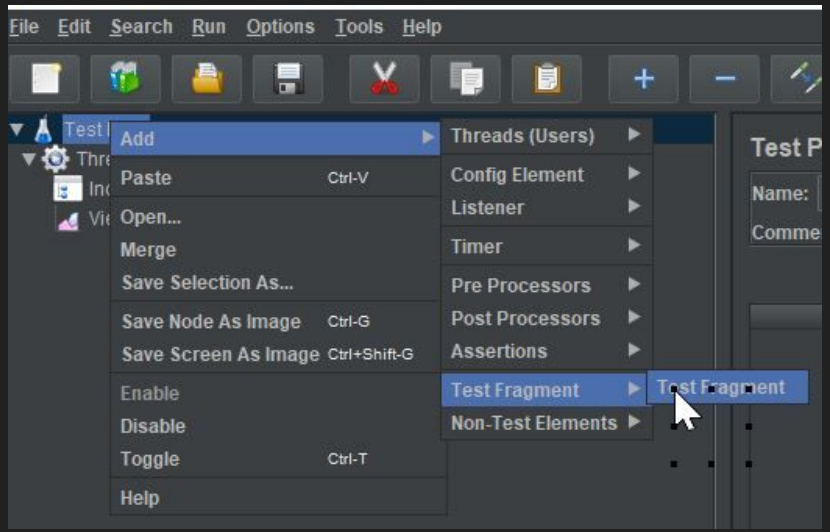
executes after a Sampler Request
to process the response data, to extract values from it



Test Fragment:

a special type of controller used with the Include Controller & Module Controller not executed unless it is referenced by either a Module Controller or an Include_Controller.

This element is purely for code reuse within Test Plans

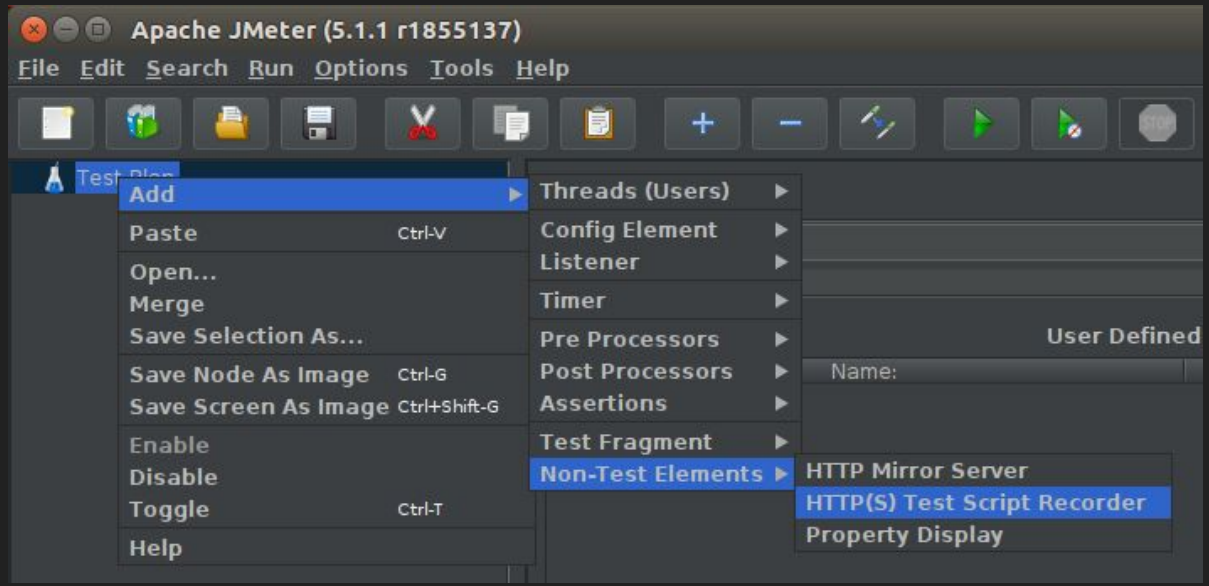


Non-Test Elements:

HTTP Mirror Server

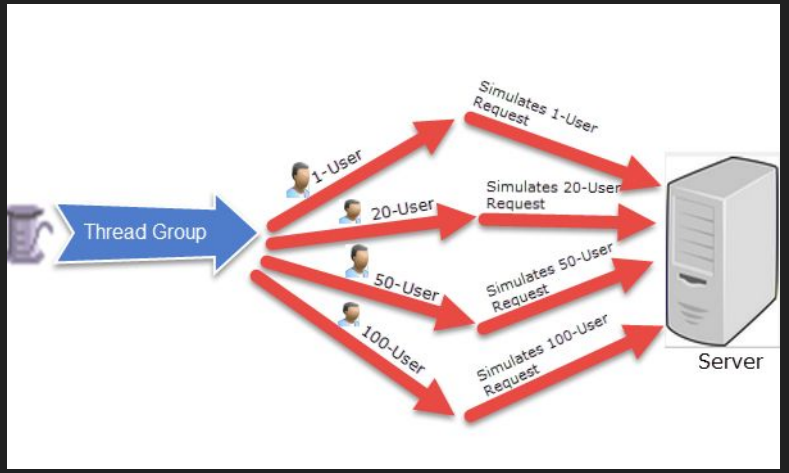
HTTP(s) Test Script Recorder

Property Display



Execution order:

Configuration elements
Pre-Processors
Timers
Sampler
Post-Processors (unless SampleResult is null)
Assertions (unless SampleResult is null)
Listeners (unless SampleResult is null)



Best practices:

1. Always do performance testing on a separate env - not used for other activities
2. Run your tests with the same infrastructure, network stats etc. - consistent results | compare with baseline
3. Create more realistic test - add think time
4. Add some ramp up, Do not start directly with 1000 users and
5. Always have a baseline to compare your test
6. JMeter checks the server performance & does not care for browser render time
7. Focus on the areas that need performance testing based on user scenario
8. Always document your results

For more information

Contacts with: [Omar Brimo || Hammad Farooq]

To download java and jmeter and Blazemeter you can see links inside the slide.

Vedio1 - Vedio2