Jmeter - Due by Raghav

① Free & open source
release ⇒ 1998

② What is Jmeter?
→ The Apache Jmeter application is open source s/w
→ a 100% Pure Java applicatn
→ designed to load test functional behaviour & measure performance.

③ Who created Jmeter?
Stefano Marzocchi

④ How Jmeter Works?
→ Jmeter mimics multiple users using the applicatn & sending request to the server.
→ It collects responses & stats from server & displayed the report of the test via tables & graphs.

⑤ What is thread group?
→ Every test plan in Jmeter has a thread group
→ this component is very important
→ here we can set the no. of users, Ramprep time, Iterations etc.

⑥ What is Sampler?
→ In Jmeter to create a request we use samplers
→ There are multiple samplers based on the different applicatn / servers / protocol
eg. for a web or web service request we use HTTP sampler.

⑦ What is execution order of components in Jmeter Test plan?
① Configuration elements
② pre-processors
③ Timers
④ Sampler
⑤ post-processors
⑥ Assertions
⑦ Listeners

Raghav → What is Jmeter?
GUI overview
Jmeter first test
Listeners
Assertions
Test script Recorder
Blazemeter Recorder
Get Data from csv file
Config Elements for HTTP
Config Elements
CMD
HTML Reports from GUI & CMD
Plugins Manager
Functions & Variables

Naveen → How to Record HTTP & HTTPS script
Blazemeter
Sampler Http request
Jmeter variables, Funcn
Jmeter logic controllers

Sdet → timers
correlations
parameterization
correlation

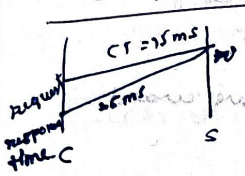
Jmeter
components
Record HTTP/HTTPS request
Timers
Samplers
Listeners
Variables
Thread
Cookie Manager
Assertions
Controllers
Regular Expressions
Data Driven Testing
Handling Dynamic Responses
Jmeter validations in Non GUI Mode
Jmeter Distributed Mode
Integration of Selenium with Jmeter
Rest API Load Testing
Monitoring Server Performance
Beanshell Scripting

Performance testing

→ Jmeter is available in the form of set of file or zip files
unzip folder → bin → executable jar file

→ PT is a practice conducted to determine how a system performs in terms of responsiveness (load time, peak time, no. of users) & stability under a particular workload.
It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as Scalability, reliability & resource usage.

* can handle load &

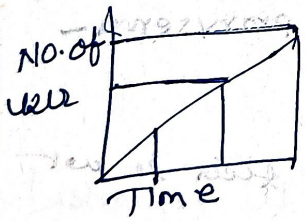


If handshake betn client & server is successful then connection has been established.

Server is available in cloud mlc or on AWS

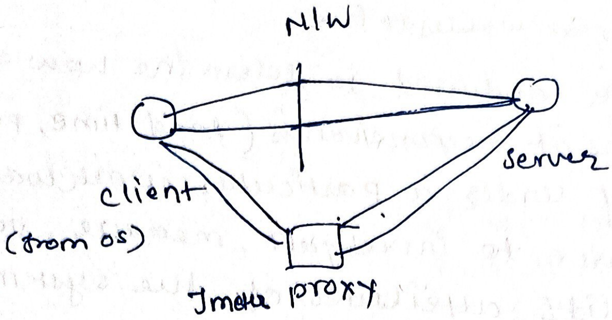
1 sec ⇒ 1 sec

① Load testing → passing users



② Stress testing → beyond the threshold value

HOW to record HTTP and HTTPs Scripts in JMeter ⇒



request send via proxy

Jmeter proxy will be in local host. port no. ⇒ 8888

→ do global setting (Jmeter proxy you have to configure over there)

adv → ① when we are going to send request to server we want to capture in jmeter UI.

we capture that ~~as~~ request, diff. internal APIs running, diff. Java scripts executed, get or post call, what exactly JSON payload we are passing in bg.

2 request → http – without any certificatⁿ
2 request → HTTPS → secured connection with SSL's certificate

⊛ How to do global configuration in Jmeter

→ internet options — connections — LAN settings — ✓ proxy server — address ⇒ localhost (8888) → ok → ok

→ open google chrome

→ uncheck all option from connections because this is not right way to doing (remove global setting)

→ firefox → setting → manual proxy → ok

firefox – options – privacy & security → view certificate

→ jmeter root ca certificate download (git repo) – download zip

→ copy paste to bin directory of Jmeter & then import in firefox.

practical ⇒ HTTP(S) Test Script Recorder (ये कैसे ~~करते~~ करना)

url pattern to Exclude ⇒ write "txt|log"

HTTP(S) Test Script Recorder → Requests filtering → url patterns to exclude → Add suggested exclude

10) Different component in Jmeter

Testplan → Threadgroup → samplers → Listeners → save → run

Thread properties ⇒ Threads ⇒ No. of virtual users

Ramp up period ⇒ how much time it will take to ramp up for all threads

No. of users → 10

20sec ⇒ Jmeter will wait ~~for each~~ thread till 20 sec to make sure that all thread are up & running.

each thread will be taken ⇒ $\frac{20}{10} = 2$ sec

↳ each & every thread will take for ramp up for all no. of threads

Loop count ⇒ No. of times that you are going to execute a particular scenario.

Sampler ⇒ HTTP request

*Sampler will be visible only in thread group.

*

Listener ⇒ it will listen & it will keep showing the data

↳ results keep showing response, request of data.

Any time in Jmeter is in msec

Config elements ⇒ parameterized some data.

latency time ⇒ that time when you getting started 1st byte of a response from server.

Latency ⇒ time to first byte.

① view Results in table

Performance Testing

Samplers ⇒ All typical operations for rest api.

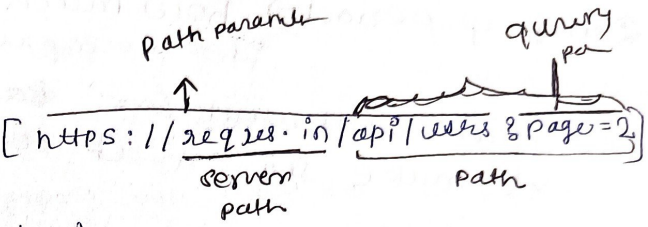
Regres.in ⇒ some dummy API

Post call ⇒ Create user

PUT ⇒ update

Delete ⇒ delete

two types of parameter
- query parameter
- path parameter



Name ⇒ GET call

protocol (http) ⇒ what type of protocol
HTTPS

Port No ⇒ see from inspect → Network tab → headers → remote address
443

request URL :- main domain url / server url

path =

Json lint ⇒ for validate Json

Goto your application
inspect → N/W

? ⇒ search

⇒ We add seperately parameter

Port No. ⇒
HTTPS ⇒ 443
HTTP ⇒ 80

Protocol
HTTPS ⇒ 443
HTTP ⇒ 80

Post ⇒
Body data in Jmeter ⇒
copy post data & paste it here.

Pass Json payload

* We need to add headers ⇒ you will get whole data (with payload)

thread group → add → config element → HTTP Header manager

copy from website ⇒ Content type – value

N/W tab - ~~Response~~ headers

then we can see header detail

⇒ ① Create all scripts then give threads

Ramp up ⇒ how much time that each thread will be taking it to start (ramp up)

Loop count ⇒ Same scenario कितनी बार execute करना है।

Authentication / HTTP Authorization :-

we can handle 2 ways basic authentication

① HTTP Authorization manager

② HTTP header manager with beanshell preprocessor

⇒ Aggregate Report ⇒ Average of all sample

median ⇒ middle line / half of the request ~~to~~ less than this time

99% line = maximum time [~~the~~

Assertions ⇒ checks on the Request / Response

① Response Assertion → any text

② Duration Assertion → कितना डाटा इतना दिखेगा request वालों के लिए fail होगा।

③ Size Assertion

④ HTML Assertion

⑤ XML JSON Assertion

⑥ XPATH Assertion

use to check response.

If you will add Assertion at test plan level then it will applicable for all samplers.

if you will add Assertion at Sampler level then it will applicable for that request only.

(Or)

Assertion ⇒ right click on request → Add → Assertion

duration Assertion ⇒ we can give durath in msec

if any sampler takes more ~~that~~ time than that ~~duration~~ we are given

whatever

if you want to check some request should not take more given that.

View results in table

⑦ Size Assertion ⇒

here we can verify size in Bytes

⑧ HTML Assertion ⇒

error thresholds ⇒ how many errors are expected.

→ can browse on html file from desktop

then whole errors/warnings will be save in give file

⑭ XML JSON Assertions

How to test API

Add → Assertn → JSON Assertion

⑤ formating & syntax for XML ⇒ XML validatn.

⑥ XML Schema

112] ⇒ Jmeter - set

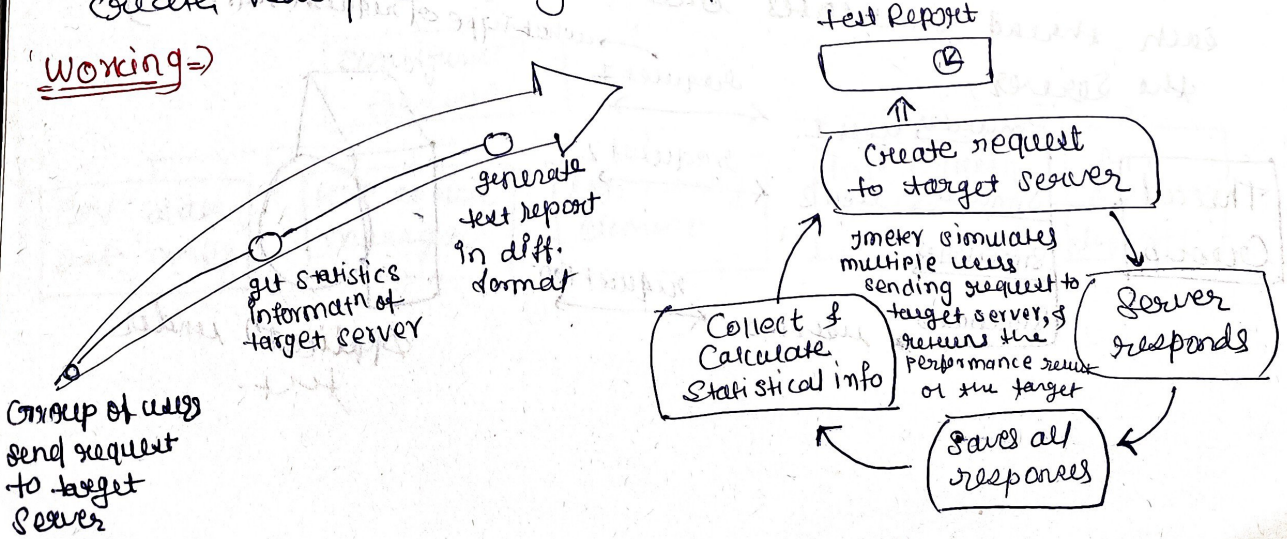
① What is Jmeter ?

→ It is performance based testing tool. purely java based SW.
→ we can easily download on other O.S.
→ It support multiple platform.
→ It is used for performance testing. we can test performance irrespective of load, stress, volume & also we can use it for web services or db or db server test or API testing.

Advantages ⇒

① It is open source license
② easy to understand (friendly GUI)
③ Platform independent (can run on mac, linux, windows)
④ Full multi-threading framework - it allows concurrent, simultaneous sampling of diff. funct group. by seperating thread group.
⑤ Visualize test Result - test results can be displayed in diff. format such as chart, tables, tree format, log files.
⑥ Easy Installation - copy & run some raw file to run jmeter
⑦ Highly extensible - write your own test
⑧ Unlimited testing capabilities - functional testing, API, web services, load, distributed
⑨ Simulat^n - support multi protocol.
create heavy load against applicat^n & test.

Working ⇒



112| ⇒ Jmeter - set

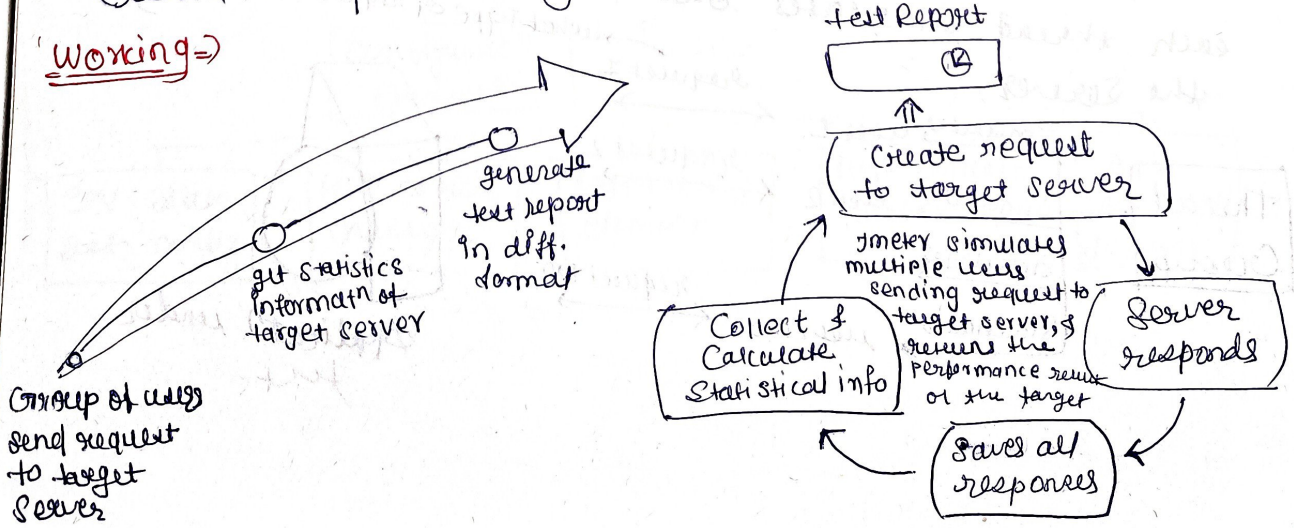
① What is Jmeter?

→ It is performance based testing tool. purely java based s/w.
→ we can easily download on other O.S.
→ It support multiple platform.
→ It is used for performance testing. we can test performance irrespective of load, stress, volume & also we can use it for web services or db or ob server test or API testing.

Advantages ⇒

① It is open source license
② easy to understand (friendly GUI)
③ Platform independent (can run on mac, linux, windows)
④ full multi-threading framework - it allows concurrent, simultaneous sampling of diff. funct group. by seperating thread group.
⑤ Visualize test Result - test results can be displayed in diff. format such as chart, tables, tree format, log files.
⑥ Easy Installation - copy & run some raw file to run Jmeter
⑦ Highly extensible - write your own test
⑧ unlimited testing capabilities - functional testing, API, web services, load, distributed/
⑨ Simulatn - support multi protocol.
create heavy load against applicatn & test.

Working ⇒



How to install Jmeter =>

How to install Java ⇒ ① Search for JDK
→ download JDK
→ set bin path in environment variable

① install Java

② google → download Jmeter (type) → Apache Jmeter → click
click on zip file → keep in c drive → bin → .bat file (for windows)
(66 mb) double click on this

→ bat file open with diff. UI.

Jmeter Elements

The different components of JMeter are called Elements.
Each Element is designed for a specific purpose.

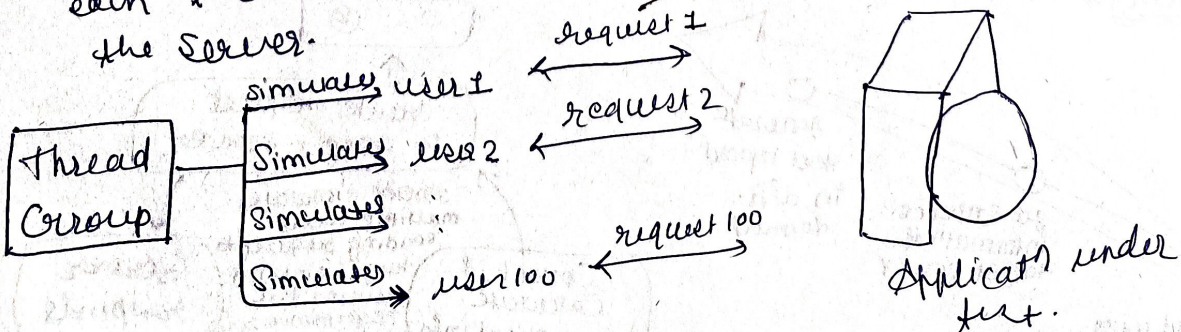
Jmeter
Thread Group

Samplers | logic Controllers | Listeners | Configuration elements | Assertions | timer | processors

Thread Group, Samplers, Listeners & Configuration

① Thread Group -
② Samplers
③ Listeners
④ Configuration

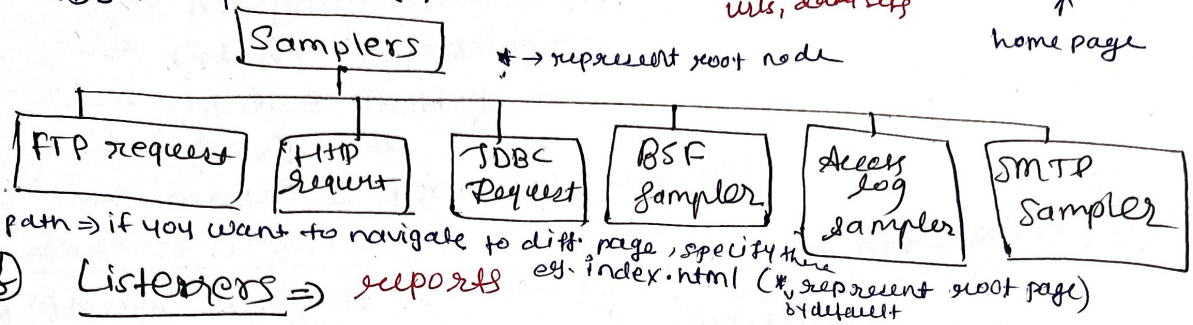
① Thread group ⇒ (no. of users) It is collection of thread. every thread represents a user, using the applicat in test.
each thread simulates one real user request to the server.



③ Samplers ⇒ type of request ⇒ added http request page

→ Samplers are diff. type of requests send by Thread group.

→ The user request could be FTP request, HTTP request, JDBC request etc.



path ⇒ if you want to navigate to diff. page, specify there eg. index.html (*, represent root page) by default

④ Listeners ⇒ reports

→ Listeners shows the results of the test execution

→ They can show results in a diff. format such as tree, graph, table or log file.

Graph Result → listens displays the server response times on a graph.

View Result Tree → show results of the user request in basic HTML format.

Table Result → show summary of a test result in table format.

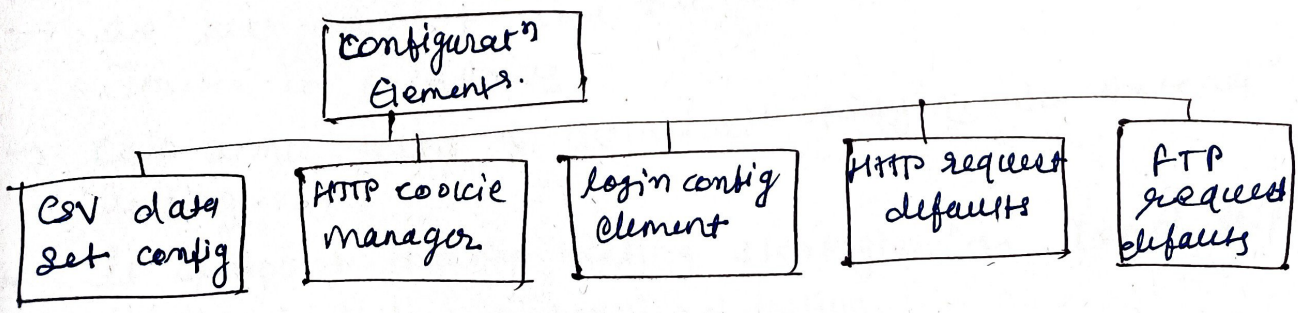
Log → show summary of a test results in the text file.

Used for Reporting. elements not gather information about the performance test used to view results/metrics of the test.

⑤ Configuration Elements ⇒ common variables which is used in samplers

→ set up defaults & variables for later use by Samplers

→ commonly used configuration elements in Jmeter.



Jmeter – GC Reddy

→ It is open source, java based
→ Ease of Use
→ Platform independent
→ Robust Reporting
→ Ultimate Testing
→ Flexibility
→ multiple protocol support

What is Jmeter →

→ Jmeter is open source testing s/w, java based application
→ It for load & performance testing
→ Apache s/w application is an american non profit corporation to support apache s/w project.
→ It was orginally built to provide open source soln for load & performance testing. It is also used to perform functional testing, applicat^n backend, load testing rest API's can be done easily in Jmeter.
→ load testing, stress testing are part (subset) of PT. endourness testing.
→ It can run on multiple platform.
→ It allows concurrent & simultaneous sampling of diff. funct^n by seporate thread group.
→ Jmeter Test result can be displayed as diff. test results such as chart, table, tree & log file.
→ we just copy & run the bat file ~~or~~ to run Jmeter in windows
→ can write tests & visualizat^n plugin to extend testing.
→ it support many testing strategies such as load testing, distributed testing & functional testing.
→ it can simulate multiple users with concurrent threads create heavy load against web applicational test.

→ It is not only perform web application test but also test data base server performance.

All basic protocol such as :—
HTTP, JDBC, LDAP, SOAP, JMS, FTP are supported by Jmeter.

→ Jmeter records all browser activity & Simulate them in web applicatn.
→ Jmeter can be integrated with Ant & Selenium for automated testing

Install Jmeter

① Install Java
② Java environment variable setup
③ download Jmeter & Run Jmeter. (from Apache)

bin → Jmeter.bat → run from bin
bin → jmeter.sh → for linux
bin → jmeter.sh → for mac

Can start Jmeter in 3 modes → GUI mode
→ server mode
→ Command line mode

How Jmeter Works?

Jmeter simulates a group of users sending requests to a target server, & return statistics that show the performance/functionality of the target server/ application via tables, graphs etc. the fig below depicts the process

Start → creations requests to

performance testing using Jmeter ⇒

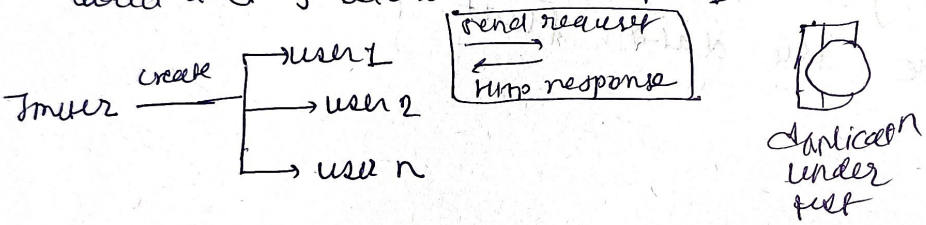
It is used to analyze overall server performance under heavy load.

It can test both static resources as JS & HTML & performance of dynamic resources such as JSP (Java server pages), Servlets & Ajax.

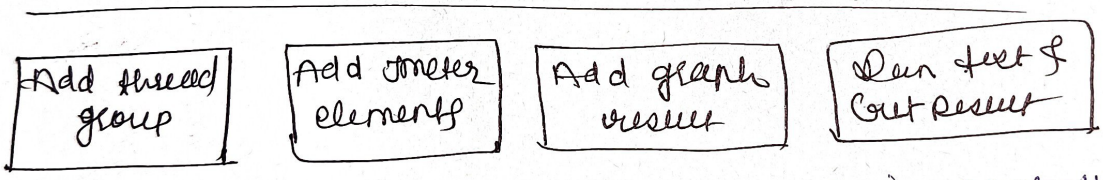
→ It can discover maximum no. of concurrent user that your website can handle.

→ It provides variety of performance report like chart form, table form, tree & log file.

→ ~~Perfor~~ Jmeter performance testing include load test & stress test. sampler



create performance test plan in Jmeter



→ consist of one or more thread groups, logic controllers, Sampler generating controller, listeners, timers, Assertions & configuration elements.

→ before creating PT for web application we need to define 2 things a normal load, avg no. of users visit our website, heavy load, maximum no. of users visit our website.

→ Step 1 → Add thread group → start Jmeter → select test plan on tree & add thread group

Step 2 → Adding Jmeter elements

Step 3 → adding results

Step 4 → Run test & get results

Limitations ⇒

use remote servers & distributed testing for larger load testing.

do not create console to create load.

are can use console for debugging purposes are to run a small load size.

GUI consumes lot of memory for heavy loads.

Limit no. of threads or users per engine.

Jmeter graph consume lot of memory.

use naming convention for all element.

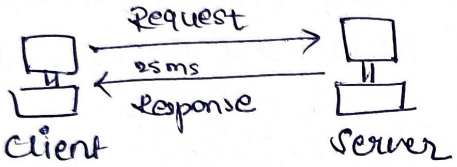
only save the data that you need.

Jmeter - Naveen ~ Important Terms

→ perf. Testing will perform on happy path (for positive scenario)

→ we nuves design framework on PT.

→ Validate response data, quality attributes



connect happen by using TCP/IP protocol by using IP address.

ResponsTime ⇒ sending request & coming back to client

response can be in the form of packets.

Bottleneck ⇒ Server will stuck

by raghav

How to record HTTP and HTTPs Scripts

Jmeter HTTP(S) Test script Recorder =)

It is element help us to ~~do~~ record our request from browser.

It can be any HTTP request.

Common performance problems

① long load time - less
② poor response time - less
③ poor scalability -
④ Bottlenecking - caused by one faulty section of code

Process ⇒ Identify test Environment

① physical test Environment - product environment
testing tools
understand details of h/w, s/w & n/w configuration
used during testing. before you begin the testing
process. so it will help testers create more
efficient tests

⇒ Performance Testing is defined as a type of Non-functional
s/w testing to ensure s/w applications will
perform well under expected workload.

⇒ The goal of performance testing is not to find bugs
but to eliminate performance bottlenecks.

→ The focus of PT is checking a s/w programs.
- Speed - whether application responds quickly
- Scalability - determine maximum user load the s/w application can handle
- Stability of s/w → determines applicatn is stable under varying loads
- resource usage

Why PT?

it is done to provide stakeholders with information about
their applicatn regarding speed stability & scalability
PT uncovers what needs to be improved before product
goes to market.

Problem ⇒ running slow while several users use
it simultaneously or sometimes inconsistencies
across diff. O.S. & poor usability.

PT → it determines whether their s/w meets a speed
scalability & stability requirements under
expected workloads.

Types of PT

① Load Testing - checks the applications ability to perform under anticipated user loads

② Stress Testing - testing an applicatn under extreme workloads to see how it handles high traffic or data processing

③ Endurance Testing - done to make sure the sw can handle the expected load over a long period of time

④ Spike testing - test the sw reactn to sudden large spikes in the load generated by the users

⑤ Volume testing - large no. of data is popolated in the db & overall sw system is monitored.
objective is to check sw applicatn performance under varying db volumes.

objective is to identify performance bottlenecks before the sw applicatn goes live - ~~Load~~ testing

objective is to identify the breaking point of an application. - ~~stress testing~~

performance Testing Process

↳ Identify Test Environment
↳ plan and design tests
↳ Identify performance Acceptance criteria
↳ Configure Test Environment
↳ Implement Test Design
↳ Execute Tests
↳ Analyse and Report

Stw PT process

test environment, production environment
what testing tools are available
understand details of the H/W, S/W & N/W
Configuration use during testing before you
begin the testing process
it will help create more efficient tests
it will also help identify possible challenges
that tester may encounter during the
performance testing procedures

⇒ plan & design tests

⇒ Identify performance Acceptance Criteria
includes goals & constraints for throughput
response times & resource allocation

identify project success criteria outside of these
goals & constraints

tester should be empowered to set performance
criteria & the goals because often the project
Specification will not include abide enough
avoid enough variety of performance benchmarks

⇒ configure Test Environment
arrange tools or other resources

⇒ Implement test design
test according to your test design co again
test design will be created according based
upon the performance requirements & the then
execute tests.

⇒ execute tests

⇒ Analyze Report ⇒ stop when bottlenecking is caused
by the CPU co then you may have
the consider option of increasing
CPU Power.

Performance Testing Metrics - parameters monitored - common parameters which you need to monitor after conducting the performance testing

① Processor usage ⇒ an amount of time processor spends executing non-idle threads.

② Memory use ⇒ amount of physical memory available to process on computer.

③ Disk time ⇒ amount of time disk is busy executing a read or write request.

④ Bandwidth ⇒ shows the bits per seconds used by a n/w interface

⑤ Private bytes ⇒ number of bytes a process has allocated that can't shared amongst other processes. these are used to measure memory leaks and usage.

⑥ Committed memory ⇒ amount of virtual memory used.

⑦ Response time ⇒ time from when a user enters a request until the first character of the response is received.

⑧ Throughput ⇒ rate a computer or n/w receives requests per second.

⑨ Amount of connection pooling ⇒ the number of user requests that are met by pooled connectn. The more requests met by connections in the pool, the better the performance will be.

⑩ maximum active sessions ⇒ the maximum no. of sessions that can be active at once.

⑪ Hit Ratio ⇒ This has to do with the no. of SQL statements that are handled by cached data instead of expensive I/O operatn. this is good place to start solving bottlenecking issues.

⑫ Hits per second ⇒ the no. of hits on a web server during each second of a load test.

⑬ Thread counts ⇒ An applications health can be measured by the no. of threads that are running & currently active.

P-T (Edureka)

PT by Rahul Shetty

① Jmeter ⇒ ① open source

file - templates - web - create

for logs ⇒ options ⇒ log viewer

options ⇒ SSL manager = if have any certificates

options ⇒ can change lang.

Tools ⇒ create a head dump → can save log file.
create a thread dump

Stop button ⇒ it will stop everything abruptly.

Shutdown ⇒ it will stop test plan gradually / gracefully
after stopping all the active threads
it will active when test plan is running.

How to create first Jmeter test

step 1: start Jmeter
step 2: create a Testplan
step 3: Create a Thread group (users)
step 4: Add a Samples (HTTP)
step 5: Add listeners
step 6: Run the Test

Jmeter by Raghav

Jmeter HTTP(s) Test Script Recorder

It is element help us to record our HTTP request from browser.

How to record your test on Jmeter
How to add & use test Script Recorder
How to add & use Recording Controller
How to use Proxy on firefox, chrome & Safari
How to Request filtering
How to use Recording Template.

Jmeter ⇒ click on Test plan

① Right click (test plan) → Add → Non Test-Element → HTTPS test Script Recorder

② Add thread group → Add → logic controller → Recording Controller

Why? ⇒ we no need to add manually this all request in our Jmeter.
We can just browse & do the all the actions & everything will be recorded
So, it will be efficient.

for 3 page → Add 3 controller

thread → Add → logic con → Recordi

① configure browser ⇒ search for proxy in PC

② Add certificates ⇒ in browser ⇒ security ⇒ manage certificates
download certificates from Jmeter - bin folder

Add cert all into then see here in tool

⇒ By using Recording template automatically unnecessary API will Remove.
File ⇒ templates ⇒ Recording or Recording with think time

Blazemeter ⇒ can save, add, edit page in blazemeter then it will save in one file we can open that jmx file in jmeter.

got pages

How to use Blazemeter to Record Jmeter tests

Step 1 ⇒ create Blazemeter Account
Step 2 ⇒ Get Blazemeter Extension
Step 3 ⇒ Login to Blazemeter
Step 4 ⇒ Record test
Step 5 ⇒ Save JMX
Step 6 ⇒ Add JMX in Jmeter & Run

Run performance test on cloud
this is cloud based

⇒ got Jmx file from blazemeter
⇒ open file in Jmeter.

How to get data from csv file

config element ⇒ csv Data set config.

→ Test plan ⇒ Add ⇒ config element ⇒ csv data set config

Step ① ⇒ Add css data set config

Step ② ⇒ Create a csv file & add csv data

goto desktop ⇒ text file save as .csv

add some data

Step ③ ⇒ Refer the csv file in jmeters csv data set config

. browse that file .

add thread group →
add java request →
add listeners

Step ④ ⇒ Refer values from csv file using syntax

$(variable Name)

Step ⑤ ⇒ Run & validate

Config Data

Testplan → Add → config elements.
elements that are executed before the sampler requests at the same level

~~run before test plan it~~.

run before the request samplers at that particular level.
if we have added config element at a test plan level then, will run before any of the requests present at a test plan level.

⇒ configuration elements can be used to set up defaults & variables for later use by samplers. Note that these elements are processed at the start of the scope in which they are found. i.e. before any samplers in same scope

① Headermanager ⇒ can add all header
② user defined variable ⇒ can add variable
${variablename}
③ HTTP request defaults. ⇒ by defaults can add
④ DNS cachemanager ⇒ if you are uses load balancer in your server then can add here

⑤ HTTP Authorizatn manager ⇒ to add username or password

⑥ HTTP cookie manager
⑦ HTTP cache manager
⑧ keystore configuratn ⇒ store for SSL certificates

Jmeter - Due by Raghav

① Free & open source
release ⇒ 1998

② what is Jmeter?

→ The Apache Jmeter application is open source s/w
→ a 100% Pure Java applicatn
→ designed to load test functional behaviour & measure performance.

③ who created Jmeter?
Stefano Mazzocchi

④ How Jmeter Works?

→ Jmeter mimics multiple users using the applicatn & sending request to the server.
→ It collects responses & stats from server & displayed the report of the test via tables & graphs.

⑤ What is thread group?

→ Every test plan in Jmeter has a thread group
→ this component is very important
→ here we can set the no. of users, Ramprep time, Iterations etc.

⑥ What is Sampler?

→ In Jmeter to create a request we use samplers
→ There are multiple Samplers based on the different applicatn / servers / protocol

eg. for a web or rest service request we use HTTP Sampler.

⑦ What is execution order of components in Jmeter Test plan?

① Configuration elements
② pre-processors
③ Timers
④ Sampler
⑤ post-processors
⑥ Assertions
⑦ Listeners

(8) what is listeners?

→ uses to show the result & metrics
→ Additionally, listeners can direct the data to a file for later use.

(9) What is Assertion?

Assertions are like checks that we can run on response received.
eg. duration, size, Response (code, text etc)

(10) What is distributed load testing?

→ Running a test on several different computers at a time.
→ makes possible simulating a large number of virtual users & involves generating a lot of traffic

(11) How to manage resource requirement, out of memory issues in Jmeter?

→ use command line (non GUI) mode
→ Disable any memory consuming listerns when doing actual performance test.
eg. View Results in Tree
Any Graphical listeners
Add ramp up time for users
Add think time betn requests

(12) what is max user load or threads Jmeter can handle?

It depends on the systems HW configurat^n
processor, memory, Allocated memory for Jmeter etc.

(13) what is co-relation in Jmeter?

process of extracting a value from a response & using it in any subsequent request.

(14) How to manage cache & cookies in Jmeter?

Using config elements like
HTTP cache manager
HTTP cookie Manager

(15) What are Timers?

By default, a Jmeter thread executes samplers in sequence without pausing

It is recommended to specify a delay by adding one of the available timers to thread group.

A time will cause Jmeter to delay a certain amount of time before each sampler.

What is Jmeter?

(1) Tool or application for performance testing.
(2) Free and open source
(3) Built with Java.

Features of Jmeter

→ Completely free
→ Ability to load and performance test many different applications/server/protocol types:
  i) Web - HTTP, HTTPS
  ii) SOAP / Rest webservices
  iii) FTP
  iv) Database via JDBC
  v) TCP
  vi) Java Objects

lot of things to do with Jmeter but I worked on web over HTTP HTTPS

Can do performance testing of a variety of applicatn
Web/API/FTP/DB/LDAP

→ Option for Recording Tests
→ In built components for adding assertions & Reporting.
→ Extensible using plugins
→ Jmeter is server side performance testing tool.

when you test your applicatn on jmeter it actually records and tests the communication with the server. so if you are using any client like a browser it will not take into consideratn it is testing server performance.

How to setup Jmeter on your system?

Step 1 ⇒ check Java is installed java -version

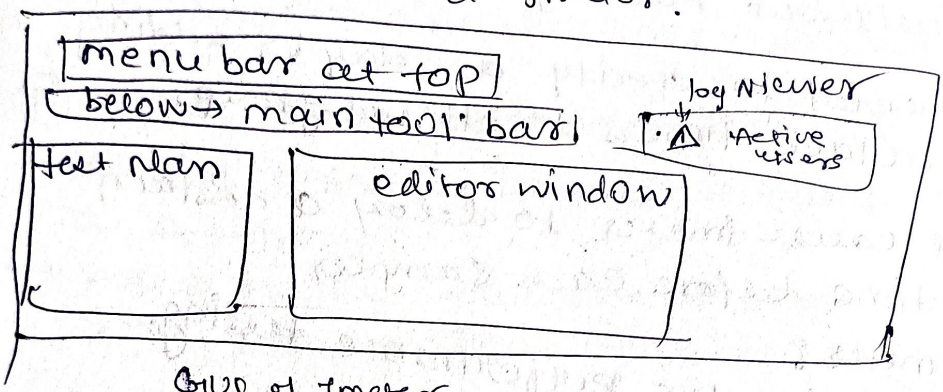
Step 2 ⇒ Download Jmeter [from binaries download zip]

Step 3 ⇒ unzip folder

jmeter.bat → use to start Jmeter

sh jmeter.sh → on cmd prompt ⇒ can start on linux or mac

Step 4 ⇒ start Jmeter.



GUI of Jmeter

How to prepare for a performance Test.

⇒ First of before start any kind of performance test then this is very very imp that you have necessary approvals, authorization and permissions to do the performance test. To do perfor. Testing or security testing it is very imp to get all perfmissions because in P.T. we will be putting so much of the load on the application. & the servers & you will be using so many virtual users so you should have permission & authorizat^n to do PT from owner of the applicatn or from your client

⇒ you should know the whole applicat^n, whole test data, the scenarios etc. you should knowing what scenarios you have to run for performance test & what is flow of scenarios

⇒ you should what is expected ~~state~~ behaviour on running the performance test with some particular amount of users what should be the performance of the applicat we can take this from stakeholder, owner, client whoever has assigned you the performance test you can ask them what is the expected outcome or what is expected performance of applicat?. how many user should i use or how many virtual users should i use to run the performance test ask for baseline.

Baseline ⇒ you have expected ~~behaviour~~ performance against the number of virtual users

for eg. for 10 virtual users → what is expected performance what is the response time average response time median, NIW time. what happen if incr. load to hundread, 50,00, 50,000 & so on.
there will be complete report in tabular or csv format.
you can compare the last time with 10 users this was the performance this time with 50 users what is the performance

HOW to prepare for a performance Test
→ Get a approvals and authorizations
→ Application and environment
→ Requirements for the test - what to test, scenarios
→ Test Data
→ Expected behaviour or performance or Baseline.

① Test plan

⇒ Test plan is a like project where all the elements of the performance test are added and stored.

create new testplan ° inside the test plan the very imp 1st element we add is the thread group

Add → thread → thread group

Name →
Action to be taken after a sampler error
• continue ○ start next thread loop ○ stop thread
• stop test ○ stop test now

Here we can can give user count, virtual users & all these details

⇒ it save as .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 and one or more Samplers

↑ this is min requirement to run any test plan

② Thread Group ⇒

⇒ A Thread Group is a collection of threads. / users
threads are virtual user
⇒ every JMeter test plan stands with this element that is thread group
⇒ each thread represents one or more user using the applicat under test.
⇒ In a thread group element, we can define the No. of users (threads), Ramp up and down, loops / Iterations

No. of threads ⇒ # 0 How long to take to ramp-up to the full no. of threads chosen

Ramp-up periods → In how many seconds should these 10 users be added.

for eg. 10 users
10 ramp upper ⇒ that means 1 user will be added every second until 10 users are completed in 10sec

loop ⇒ if you want to iterate then give loop count

Add → Threads → ① Thread Group
② setUp Thread Group
③ tearDown Thread Group

SetUp Thread group ⇒ it will run before the actual thread group.

teardow ⇒ it will run after the actual thread group is executed.

Controllers

Jmeter has two types of Controllers -
Samplers and logical Controllers

→ they are element to process the Request so its called as controllers.

Thread → add → Sampler → there are so many samplers available in Jmeter we can use it based on applicatn

I used it for web applicatn so we need to Select HTTP request ~~as~~ Sampler.

~~for~~ if yours are running FTP then can Select FTP request Sampler.

Similarly there will be JDBC sampler for database & so on.

Based on Applicatn, we have samplers in Jmeter & these are actual elements that run the request.

Samplers ⇒ Samplers tell Jmeter to send requests to a server and wait for a response

(it request add as it does) They are processed in the order they appear in the tree.

If you want to change sequence then can use controllers

⇒ each sampler (except flow control Action) generates one or more sample requests.

use to delay time or give pause request duratn is in-mill

Logical Controller ⇒

→ the elements that determine the order in which the samplers are processed

→ let you customize the logic that jmeter uses to decide when to send a request

Basically it will control order or logic

Thread group → Add → logic controller →

there are so many logical controller are available in jmeter

if you want to add some condn before running your sampler request. we have transaction controller, loop controller, while, if, critical section, foreach, include, Interleave, once only, Random, Random order, Recording, Runtime, Simple, Throughput, module, Switch

determine the order in which the samplers are processed.

Listeners ⇒ Reporting element

Thread → Add → listeners →

View Results Tree, Summary Report, Aggregate Report, Backend listener, Aggregate graph, Assertion Results, Graph Results, Comparison Assertion visualizer, generate Summary Results, JSR223 listener, mailer visualizer, Response time graph, Save Responses to a file, Simple Data writer, view results in table, Beanshell listeners

Some are graphical, tree, tabular listeners.

- Stop button ⇒ abruptly stop the session without waiting for the threads to complete
- Shutdown ⇒ it will gracefully end the test it will wait for all the currently running threads to complete the execution & then it will shutdown.

⇒ components that show the result of the samplers
Tree | Table | Graph

⇒ also provide means to view, save and 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

How Run test on cmd prompt

Step 1 ⇒ open cmd prompt
Step 2 ⇒ Go to the locatn of jmeter bin folder
Step 3 ⇒ Run the command

jmeter -n -t "locatn of jmx file" -l "locatn of result"

-n ⇒ non gui mode
-t ⇒ locatn of testfile
-l ⇒ log the results

To minimize the amount of memory needed, use the Simple Data writer and use CSV format.

Assertion

⇒ Assertion = check or validation
check Actual = Expected
processed after every sampler in the same scope.

Thread group → Add → Assertion

Response, JSON, Size, JSR223, Compare, Duration, HTML, JSON JMESPath, MD5Hex, SMIME, XML, XML Schema, Xpath, Beanshell

based on our applicatn, based on Responses you are getting we can add this assertion

Config Elements

can be used to set up defaults and variables for later use by samplers.

Thread → Add → config element

CSV data set config, HTTP header manager, HTTP cookie manager, HTTP cache manager, HTTP request defaults, Bolt connectn configuratn, Counter, DNS cache manager, FTP Request defaults, HTTP Authorizatn manager, → JDBC connectn configuratn (set connectn to db) Keystore configuratn, Java request defaults, Config element, LDAP Request defaults, login Simple config element, Random ~~element~~ variable, TCP Sampler config, User defined variables.

⇒ config element runs first
if any config element in your test plan they will be executed first & then you will have other elements

⇒ processed/it will be executed as per the scope it is present

Timers

Timers are to add some delays or pauses if it is recommended that you always add some think time & some delays in your test so that it can mimic a real world scenario

thread group → Add think time to childdren

thread group → Add → Timer →

Constant timer, uniform Random, Precise throughput, constant throughput, Gaussian Random, JSR223, poisson Random, synchronizing, Beanshell

⇒ to add delay (pause) before sampler request
⇒ By default, a 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 → accordingly add the pause delays
⇒ it can be added as children of samplers or controllers

preprocessors ⇒

preprocessors are the elements that will be executed before the sampler request

post processors will be executed after the sampler request.

executes prior to a sample Request.

modify the settings of a sample request or update variables

Thread group → preprocessor →

JSR233 preprocess, User parameters, HTTP link parser, HTTP URL Re-writing modifier, JDBC preprocessor, Regex User parameter, sample timeout, Beanshell preprocessor

thread group → Add → post processors →

CSS selector Extractor, JSON extractor, JSON JMESpath extractor, boundary extractor, Regular expression extractor, JSR223 postprocessor, debug post processor, JDBC post, Result status Action Handler, XPath extractor, XPath 2 extractor, beanshell postprocessor.

post processors ⇒

execute after a Sampler Request

to process the response data, to extract values from it

Test fragment ⇒

⇒ 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 plan.

with include controller & only use for

Preprocessors ⇒

preprocessors are the elements that will be executed before the sampler request

post processors will be executed after the sampler request.

executes prior to a sample Request.

modify the settings of a sample request or update variables

Thread group → Preprocessor →

JSR233 preprocessor, User parameters, HTTP Link parser, HTTP URL Re-writing modifier, JDBC preprocessor, Regex User parameter, Sample timeout, Beanshell preprocessor

thread group → Add → post processors →

CSS selector Extractor, JSON extractor, JSON JMESPath extractor, boundary extractor, Regular expression extractor, JSR223 postprocessor, debug post processor, JDBC post, Result status Action Handler, XPath extractor, XPath 2 extractor, beanshell postprocessor.

Post processors ⇒

execute after a Sampler Request

to process the response data, to extract values from it

Test fragment ⇒

⇒ 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 plan.

use with include controller & only use for use with test plan

Non test Elements ⇒

testplan → Add → Non test element → HTTP mirror server
HTTP (s) Test Script Recorder
Property Display

HTTP test script recorder is used to record our request. if you don't want to ~~add~~ manually add request or want to record HTTP ~~or~~ test script

property display ⇒ to see all the properties of your system

mirror server ⇒ if you want some data & mirror the servers from & get the data from there you can use HTTP mirror server.

Execution order ⇒

1. configuration elements
2. Preprocessors
3. Timers → calculate pause time & delays
4. Sampler
5. post - processors
6. Assertions
7. Listeners

} execute only when you have sample result

- setup environment ↓
do not hamper, you do not interfere with any other activities, do not bring down the servers

Best practices ⇒

① Always do performance testing on a separate env. - not used for other activities

② Run your tests with same infrastructure, network state 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

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 needs performance testing based on user scenario

⑧ Always document your results.

Non-functional Testing ⇒ it is to determine the system performance i.e. responsiveness, stability, reliability and scalability under a particular workload.

Ramp-up is the amount of time it will take Apache Jmeter to add all test users (threads) to test execution.

How long it will take to Jmeter to start execution of all the threads.

for eg.

- 1000 target threads with 1000 seconds
  ramp-up ⇒ Jmeter will add one user each second
- 1000 target threads with 100 seconds
  ramp-up ⇒ Jmeter will add 10 users each second
- 1000 target threads with 50 seconds
  ramp-up ⇒ Jmeter will add 20 users each second.

⇒ PT determines how the stability, speed, scalability and responsiveness of an application holds up under given workload.

It determines whether SUT meets speed, scalability and stability requirements under expected workloads.

To understand PT we need to think of these two metrics – latency and throughput.

99% line ⇒ 99% request were under 660 msec

throughput ⇒ no. of request per sec matrix

I.W que

① How will you do end to end P.T

①– what scenario that has to be tested? (you should get all requirement clearly)

② what is the expected user Load?

③ do we need geographical distribution?

do we want to put load on the server from different locatⁿ or different geographics or a single geography

→ you should get everything very clearly stated & you should understand all the requirements very clearly

→ ask for PT environment

never ever do any PT on any existing environment which is already being used for other activities & if you do that it can compromise the N/W, it can bring down the entire infrastructure & it can hamper other activities

so always do PT on seperate environment

ask to create seperate environment for PT

once have all the requirements clearly stated down written down & have seperate environment

start PT by creating test plan → plan for the scenarios that are in scope & once you create these scenarios. you will run it with single user & check everything is fine then you should get this plan approved from the concerned person only after get a approval you should inc the load then you should inc load gradually then you should start with max load for eg. if you want to do our test with 10,000 users do not put load of 10,000 users directly go gradually 10 – 100 – 200 & so on

then should capture all the results, you should have a baseline created

Actual Result → Baseline ⇒ you should have a baseline result or you should have some number against which you can analyze or complete your results

this request should respond within 0.5 sec you, so you have compare all you results against this baseline

In Row ⇒ you have to show your experience, your skills, or your knowledge you have to ans to the best of your abilities

Why PT? needs
explain to client →
take eg of how your previous projects help the clients to mitigate the issues in product & how they were able to scale up their applications with a good PT & then you can also take eg of what can be the consequences if you do not do a proper PT
one of eg I can give you is in 2014 there was an online shopping website flipkart that announced some discounted a sale for few days called big billian days & when it started there was so much load on the servers that the site actually crashed so if you want to mitigate or avoid these kind of issues then PT is very very necessary.
In this way you can show credibility on PT.

How to test 1000+ user →
first of all set your req at ease & then take down all the requirements so you should be very very clear on the requirements what should be the actual load & what should be average load
minimum load, maximum load
what is the geographical distribution if it is required.
what are scenarios to beat undertaker for the test
All environment created & requirements very clearly stated then start with single user test then you can test

PT is a type of testing to ensure SW applications will perform well under the particular workload.

Why ? ⇒ what if 10,000 users use at a time

Server side testing
for traffic control
all defects of performance revealed by PT.

① develope scripts then apply load ⇒ 1000, 10K Shrs

Jmeter ⇒ ① open source (free of cost)
② cross platform support (built on java)
③ can work with all OS (mac, windows)
④ Scripting isn't essential to learn Jmeter
⑤ Jmeter GUI is more user-friendly

download ⇒ binaries ⇒ select OS
download java ⇒ download JDK

test plan ⇒ plan how do you want to test this.
Can work on only one project.
⇒ It consist of all actions and components you need to execute your performance test script.

Workbench ⇒ rough folder It can be taken as a practice area or temporary storage as the components of workbench are not saved along with a test plan.

~~A most important component of workbench are not saved along with the test plan~~

HTTPS Testscript Recorder which can record the script directly & tester can put the load on those later on.

Record and play back

test - go and perform actions in browser

testplan → add → nonTest Element (only for recording) → test Script recording

⇒ record all actions

Name ⇒ name of recorder

जिस port से listen करना है वो डालो

Jmeter automatically captures domain [HTTPS domains]

target controller → workbench ⇒ HTTP(S) test script recorder

Request filtering →

↑o test we need requests and responses

Gif, banners

We can exclude from recording which we don't want

URL patterns to Exclude → add Suggested Excludes

Need to do settings in → ① Jmeter
② firefox browser

We can track actions from browser with the help of port No.

recording start करने से पहले browser में action perform करो

before your record test need to import one certificate in browser

browser → certificate (it is must to see Https website & record)

Test - go & perform actions in browser -

to test we need requests & responses

Gif, banners

① Jmeter

② browser

Https ⇒ that Https website needs certificate - Jmeter

* Recording with chrome using Blaze Meter Plugins

Blaze Meter → provide plugin available in chrome → add to chrome as extensions

initially blazemeter will record all actions in jmx file

then it will import in jmeter.

then we no need to use script recorder or set proxy in browser.

export int from blazemeter plugin

blazemeter will not record unnecessary script like test script recorder.

books ⇒ packtpub.com
Safaribooksline

Thread Group

A thread group is a set of threads executing the same scenario.

can place load on jmeter.

50 user hit & compute all request

Rampup ⇒ In how many seconds, you want to be users on that site.

user ⇒ 50
Rampup ⇒ 5
↑
user to create real time situation

In 1 sec 10 user will be executing flow
2nd sec 10 user
3rd sec 10 user
4th sec 10 user
5th sec 10 user

In real time, No 1000 users will click on button

loop count ⇒ if you want to repeat the flow 5 times then it will be 5

till how much time you want to iterate give time in Scheduler configuration → duration

* Recording with chrome using Blaze Meter Plugins

Blaze Meter → provide Plugin available in chrome → add to chrome as extensions

(*) initially blazemeter will record all actions in jmx file
then it will import in jmeter.
then we no need to use script recorder or set proxy in browser.
export it from blazemeter plugin
blazemeter will not record unnecessary script like test script recorder.

books ⇒ packtpub.com
Safaribooksline

Thread Group

A thread group is a set of threads executing the same scenario.
can place load on jmeter.
50 user hit & compute all request
Rampup ⇒ In how many seconds, you want to be users on that site.

user ⇒ 50
Rampup ⇒ 5
↑
user to create real time situation

In 1 sec 10 user will be executing flow
2nd sec 10 user
3rd sec 10 user
4th sec 10 user
5th sec 10 user

In real time, No 1000 users will click on button

loop count ⇒ if you want to repeat the flow 5 times then it will be 5

till how much time you want to iterate give time in Scheduler configuration → duration

Listeners

⇒ it listens to your executions

Add → listener

commonly used ⇒ View results tree
Performance metrics — Aggregate Report
Graph Results

① View Result tree ⇒ Verify all requests are passed then check in aggregate report

② checkpoint ⇒ check with single user
created jmeter script
Thread group - 1
view results tree -

You may get 200 status code - but response may wrong

eg. Add Employee ⇒ first time response ⇒ employee added
2nd time response ⇒ employee already exist.

Challenges face during PT ⇒

when I try to run result, result tree shows pass but response is wrong.
In this way wrong request show in performance metrics

listeners to monitor the load testing results ⇒

Jmeter recorded in headless mode.

Aggregate Report ⇒

Samples ⇒ ~~no~~ this many no. of user hit the request
(no. of users hit that specific request = 2135)

Average ⇒ average time each request hit.
It is average time taken by all the Samples to execute specific label. ⇒ 129ms

when 2153 samples are hit 129msec is an avg time taken by each sample to get can response

min ⇒ min time taken by user to get an Response
one of the user get response early

max ⇒ one of samplers took maximum time to get response

user को max time लगकर response मिलते
<u>worse performance</u>

Error % ⇒ can ignore if it is 3% or 4%.
percentage of failed request per label

Throughput ⇒ It is no. of request that are processed per time unit (seconds, minutes, hours) by the server.

When 2000+ users hit the request at a time, that time 72.3/sec request sent in one second.

Stamina of server can know through throughput.

→ time taken by server by handling or processing users. gives server capability.

min ⇒ it is time taken to get the message response back to user.

⇒ this time is ca

⇒ larger throughput is better.

With 100 users of load, throughput is 123.7 req/sec for reserve.php

benchmark will be given by client.

Different types of listeners & their usage in gathering performance metrics :-

90% ⇒ 90% of the samples took no more than this time. The remaining samples took at least as long as this (90th %)

median ⇒ it is time in the middle of a set of samples result. it indicates that samples took no more than 50% of the samples took no more than this time. i.e. remainder took at least as long.

Standard deviation

This shows the set of exceptional cases which were deviating from the avg value of sample response time. The lesser this value more consistent the data.
SD should be less than or equal to half of the avg time for a label.

ideal graph ⇒

① deviation should decrease
② throughput should increase

Additional plugins for simulating real time load ⇒

download the jar from ⇒ https://jmeter-plugins.org/wiki/pluginsmanager/
search ⇒ jmeter plugins → download plugin manager

Jmeter → options → plugins manager

① download plugins manager jar file & put it into Jmeter's lib/ext directory. then start Jmeter & go to "options" menu to access the plugins manager.

concurrency thread group & ultimate thread group

Custom thread group plugin → helps us to simulate real time load in MIC

After adding new thread group
test plan → Add → thread group → concurrency thread group

Ramp up time ⇒ in this time All request will be on server
target concurrency ⇒ 60
Ramp up ⇒ 30
Ramp up steps count ⇒ 3 (3 steps)

30 sec में 60 users or hit आएंगे

$\frac{60}{30} = 2$ sec

thread group → ultimate thread group

diff ⇒ start thread count → 50 user
initial delay ⇒ 5 sec (after 5 sec
Startup time ⇒
Hold load for sec ⇒
Shutdown time

if clients are outshore they may ask to do this

✱ Why HTTP cookie manager ✱

Usage of cookie ⇒

login to fb ⇒ one session cookie will create which will stored by browser.
it will uniquely identify for particular user profile.

HTTPS → stateless → it will not remember previous request

close browser → then login to fb → it will not ask you again to login

Jmeter will not store all cookies like browser

Http cookie manager ⇒ it will store all cookies like browser

Authentication, cookie storage then add HTTP cookie manager

Validate Jmeter tests

Add-Assertion ⇒ if response in xml then mainly used

① ~~JSON~~ XPath Assertion. ② XML

~~Shortcut~~ if there is web request, we mainly put assertions for Response Assertion, size, duration & HTML Assertn

Rest API ⇒ JSON Assertion

① Response Assertion ⇒

~~text~~ Response, Response code, Response message, Response headers, Request headers, url sampled, document, ignore status, request data.

Response Assertion

① ~~Text~~ Response
② Response Code

Apply to ⇒

if you want to put validation on child sample then choose "sub-samples only"

Size Assertion ⇒

→ How much size in bytes the response is generated
→ how much process have been happened in server side
Response is not with complete size of byte due to heavy load, it may not be return full size of byte

Size Assertion ⇒
Response size field to test ⇒
- Full response
- Response headers
- Response body
- Response code
- Response message

Use when there is heavy load in applicatn.

Duration Assertion ⇒

How much time request is taking to generate response

duration in msec ⇒ 30 → maximum benchmark
→ maximum it should be 30sec to generate

max કેટલા time લેશે request ની આવો રિસ્પોન્સ આપવા માટે

duratn + size good to check performance

if you want to verify script

First of all there should be stable results, so validate Response Assertn. then can put size + duratn Assertn

test should be right

Controllers =>

How controller help to record

① transaction ② loop ③ if while ④ Runtime
⑤ Recording ⑥ module

① Recording controller ⇒ Specifically dedicated controller where all recording will be there.

②⑤ Transaction controller ⇒ (transaction reporting)
Can see any time at once in Aggregate Report
require

③ Simple controller ⇒
It's just a container to place a call request
can place at seperate folder that set.
ce

④ module controller ⇒ dos reuse the code
Which module you want to execute.
All modules will be display here.
it will increase Sample count in Aggregate report

⑤ Interleave Controller ⇒ container
One Sampler per iteration is executed in top to bottom order.
Interleave के अंदर कोई request दो होर thread group में loop count = 2 तो दो request एक ही बार execute होगी।
so, one sampler per iteration.
eg. search can be diff.

Runtime controller ⇒
it will keep on hitting the server for given time.
help you to put regressely load on server (
when some module peak time for eg. if there is sale
breaking at
or more no. of user will going to hit that module
then use this controller.

"It controls the execution of its samplers / requests for the given time"

Timers ⇒

→ customer click 50 times in 1 sec → rarely possible
——"—— 50 times in 60 sec → possible

if you want to give some real time mimic
can give some sleep btn threads
make your thread to sleep for 2-3 sec
constant timer

Gaussian timer ⇒ dviatn ⇒ it gives to every thread.
keep on changing, random value
given random timer

* Constant throughput timer *

target throughput ⇒ how many bytes are returning per min.

Data Driven testing

datadrive into application ⇒

add - config ele - csv data set config

Random Controller ⇒ (only one execute not all request)
It plays only of its children samples picking it randomly.

If controller ⇒
written using beanshell
"${store}" = "11" #condn false will not display.
Sampler or controller may change based upon ~~co~~ condn of controller.

Loop controller ⇒ we can loop in for specific modules

Regular expression extractor

Webtours – the-internet.herokuapp.com/login
blazedemo.com
~~web~~ mercury webtours → newtours.demoaut.com
tesplan – add → Non test recorder – Http script recorder
for dynamic purpose we use regular expression.
after processing by this jmeter
add – post processors → regular expression extractor

Apply to →

Scan entire response body of login then extract username

Create variable in Http request →
username $ (Username)
, Regular expression "copy from response & paste here"

execution control ~~search~~ See it need to ~~check~~ search in response body

Regular expression में जो देंगे वो response body में check करेंगे।

.+? ⇒ can have any character (achieve dynamic) (any no. of character)

literals → regex.com

? ⇒ matching set from response

.+? ⇒ get me anything

जो भी variable में save करना है उसके part को bracket में रखो और template field में mention करो ⇒ 1
↑ 1st bracket

match no. ⇒ 0 → any match from given
1 → give first match

Combining multiple expression into single extractor

Name of created variables : username

Regular Expression : Enter (.+?) for the username.

Templates (i) : when i capturing group no. starts at 1 : 1
↑ 1st bracket की random value replace होगी

Match No. (0 for Random) : 1

default Value :

Test Plan → Add → Sampler → debug Sampler
all the variable will display here

* for each dynamic variable, we can create new regular expression OR

grab whole text from response to regular expression

R.E → Enter (.+?) for the username and (.+?) for the password

From debug Sampler → you can see all the global variables see there which variable having both variables

Authentication की http request में variable ~~दिये हैं~~ हम पढ़ेंगे ?