JMeter

1. What is JMeter.
2. JMeter Advantages.
3. Working of JMeter.
4. JMeter version history.
5. How to install JMeter on windows.
6. JMeter Elements
7. Create First JMeter Test
8. Timers in JMeter
9. Controllers in JMeter
10. Simple Controller
11. Random Controller
12. Random Order Controller
13. Interleave Controller
14. Modular Controller
15. Test Fragment
16. Include Controller
17. Throughout Controller
18. Recording Controller
19. Overview on performance testing
20. What is JMeter –

* JMeter also known as “Apache JMeter” is an open – source, 100% java-based application with a graphical interface.
* It is designed to analyse and measure the performance and load functional behaviour of web application.
* JMeter was originally written and developed by ‘Stefano Mazzocchi’ of the Apache software foundation.

1. JMeter Advantages –

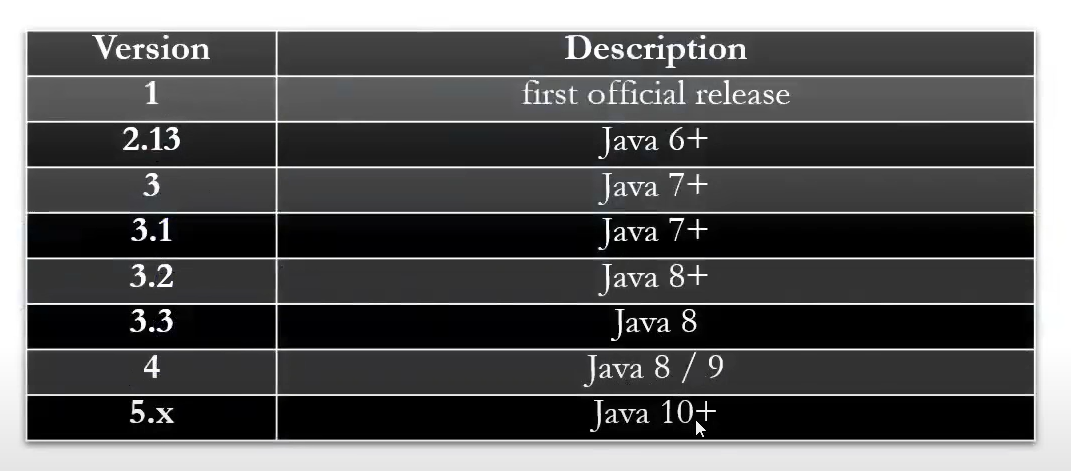
* Free and open – source
* Easy Installation
* User – Friendly GUI
* Supports various testing approach.
* Supports multi – protocol.
* Platform Independent.

1. Working of JMeter –

Diagram

Description automatically generated with medium confidence

1. JMeter version history –



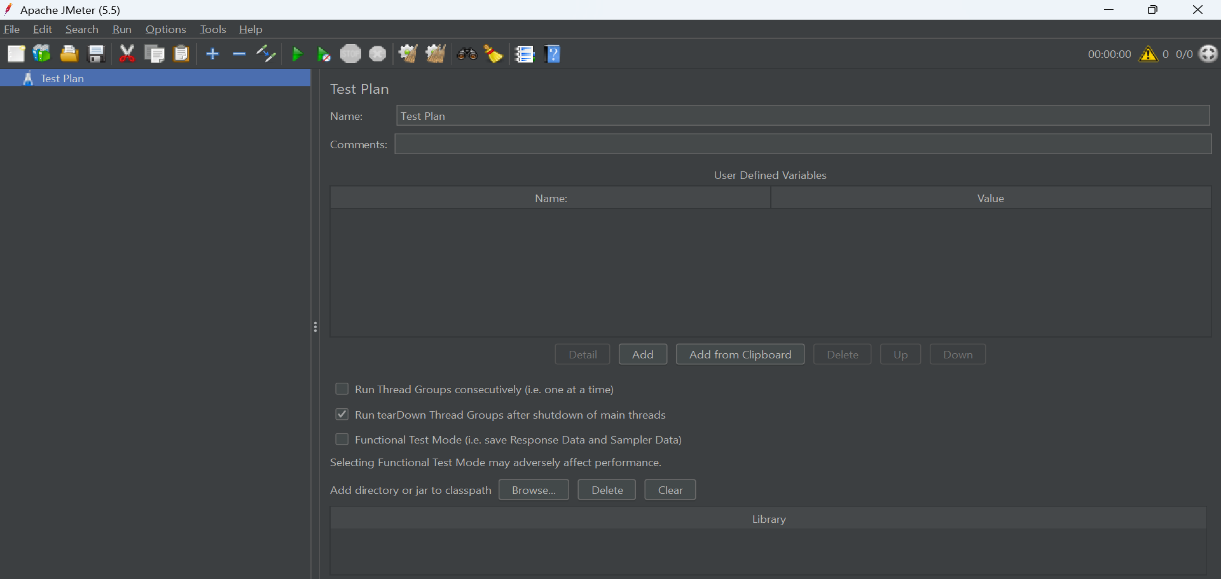
1. JMeter Installation –

* Check Java is available using command prompt.

Java -version

Java must be installed.

* Go to the Google and type ‘JMeter download’ and click on ‘Download Apache JMeter’.
* Select the ‘Binaries’ and download the Zip file.
* Unzip the zip file, open bin folder, and click on ‘jmeter – windows batch file’.
* It slowly opens the command prompt and opens the JMeter application.



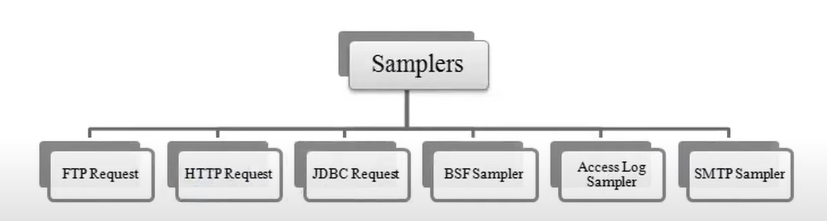
1. JMeter Elements –

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

Graphical user interface

Description automatically generated with medium confidence

* Thread Group –
* Thread group is a collection of threads. Each thread represents one user using the application under test.
* 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 –
* Samplers are different types of requests send by Thread Group.
* The user request could be FTP request, HTTP request, JDBC request…etc.



* Listeners –
* Listeners shows the result of the test execution.
* They can show results in a different format such as a tree, table, graph, or log file.
* Graph results listeners display 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.
* Configuration Elements –
* Setup defaults and variables for later use by samplers.
* Commonly user configuration elements in JMeter are:

Diagram

Description automatically generated

1. Create First JMeter Test –

Step – 1: Start JMeter.

Step – 2: Create a Test plan.

Step – 3: create a Thread Group (users).

Step – 4: Add a Sampler (HTTP).

Step – 5: Add Listeners.

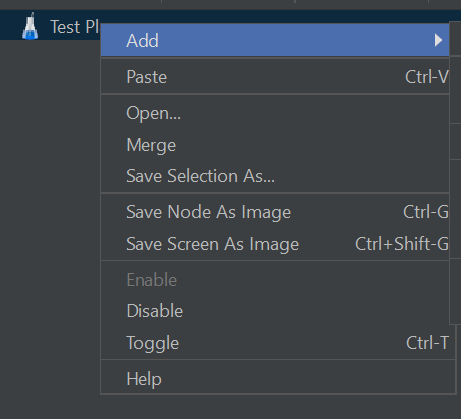
Step – 6: Run Test Plan.

Step – 7: Save Test Plan.

* Test Plan –

Test Plan is a container which contains a lot of things like all the elements of JMeter will be created here under test plan and the entire project will be saved as a part of the test plan.

* Creating a Thread Group –
* Right click on ‘Test Plan’.



* Click on ‘Add’ and select ‘Threads (Users)’.
* Click on ‘Threads’ and it shows –

Graphical user interface, text, application

Description automatically generated

Thread Group – Represents the number of users.

Name – Name of the Thread Group.

Number of Threads (users) – How many numbers of users should perform this test.

By default, one will be there we can increase this number.

Ramp – Up – Period (in seconds) – Suppose if the ramp – up – period is three seconds, and number of threads is five, then every user will send the request after three seconds of interval.

Loop Count – How many times the user should send the request. If we click on ‘Forever or infinite’, its keep on going on.

* Add a Sampler –
* Click on ‘Users’ and select ‘Add’.
* Click on ‘Add’ and select ‘Sampler’.
* Click on ‘Sampler’ and select ‘HTTP Request’, then it opens a ‘HTTP request’ window.

Graphical user interface, text, application

Description automatically generated

* Add Listeners –
* Click on ‘Users’ and select ‘Add’.
* Click on ‘Add’ and select ‘Listener’.
* Click on ‘Listener’ and select ‘View Result Tree’.

Graphical user interface, application

Description automatically generated

* And do the same process and select ‘View Results in Table’ under ‘Listener’.

Graphical user interface, application, Teams

Description automatically generated

* Run Test Plan –

Once you run the ‘Test Plan’ the output in the ‘View Results Tree’ and ‘View Results in Table’.

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

1. Timers in JMeter –

* When we apply load or stress testing, by default it takes no delay in time, sometimes it leads to the overloading.
* So, Timer element can be added to a test plan to apply wait between each sampler / request.
* Various Types of Timers in JMeter –
* Constant Timer
* Uniform Random Timer
* Gaussian Random Timer
* Bean shell Timer
* BSF Timer
* JRS223 Timer
* Constant Timer –
* Step – 1: Create one ‘Thread Group’, under thread group create three ‘HTTP requests’ and one ‘View Results in Table’.
* Add any website URL in three request pages and run the test plan.

A screenshot of a computer

Description automatically generated

Under the above picture, the three requests run in one second, sometimes it overloads and fails. Then we need to add ‘Timers’.

* We can add ‘Constant Timer’ at Thread Group level and at request level and fix time in milliseconds.

Graphical user interface

Description automatically generated

When we run the above picture, all the requests compile by taking five seconds of interval.

* If we add another timer at first request, the first request takes ten seconds and remaining takes five seconds.

Graphical user interface, text

Description automatically generated

* Uniform Random Timer –
* Random Delay Max

Constant Delay Offset

* Formula –

-0.X\*Random Delay Max + Constant Delay Offset

* X: 0-9
* Example:

0.X\*100+0

0-99 milli sec

* We can add ‘Uniform Random Timer’ at both thread group and request level.

Graphical user interface, application

Description automatically generated

When it runs, the request gets compilers by the random time.

1. Controllers in JMeter –

* Logic Controllers will decide ‘When and How’ to send a request to a web server.
* Logic Controller Types –
* Critical section controller
* If Controller
* Loop Controller
* For each controller
* Loop Controller –

Loop Controller makes the user request run a specified number of or run forever.

Graphical user interface, text

Description automatically generated

* If loop count is three at the ‘Thread Group’, all the requests run in three times.

Graphical user interface, text, application

Description automatically generated

* But we want to run the ‘Page1’ for five times and remain should same, then we need to create a loop controller and fix loop count as five.

Graphical user interface, application

Description automatically generated

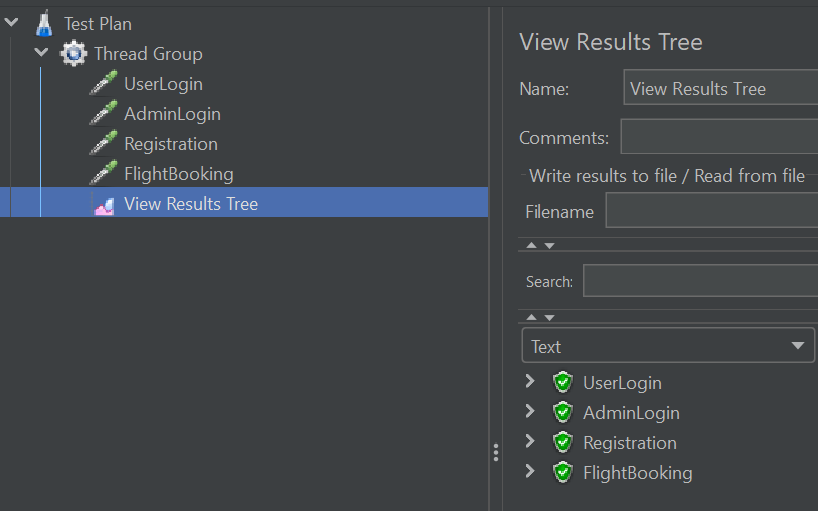
Drag and drop the ‘Page1’ request in loop controller and move the loop controller to the same position.

A screenshot of a video game

Description automatically generated

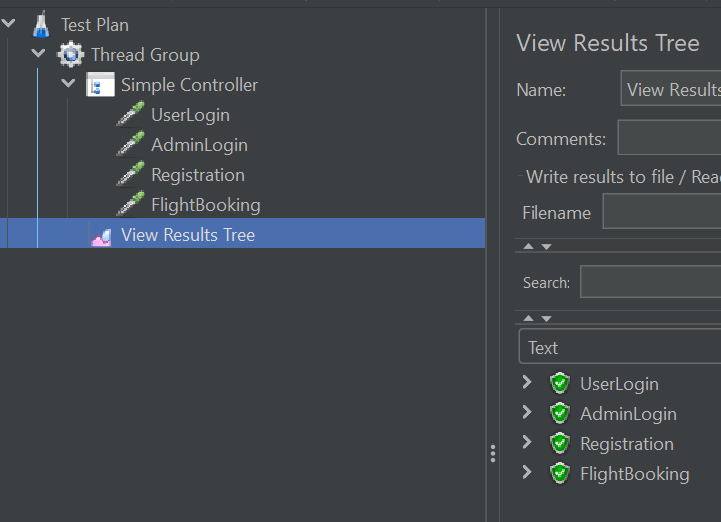
1. Simple Controller –

It is the container, where multiple requests run in a sequential order.



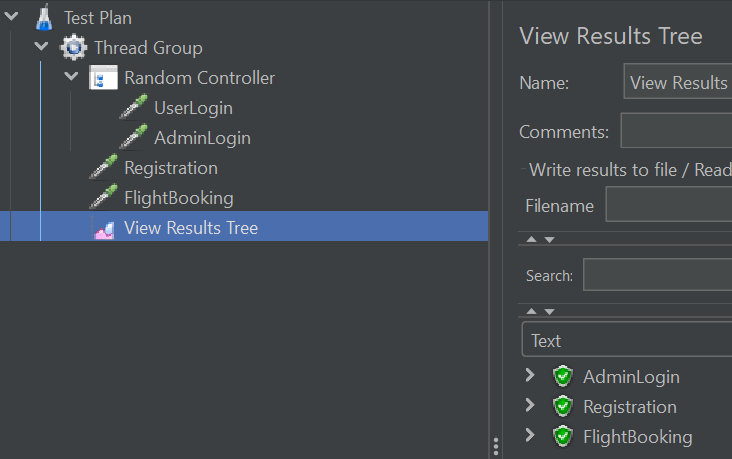
Here, requests are running under the ‘Thread Group’.

Now, we will see the requests are running in ‘Simple Controller’ –



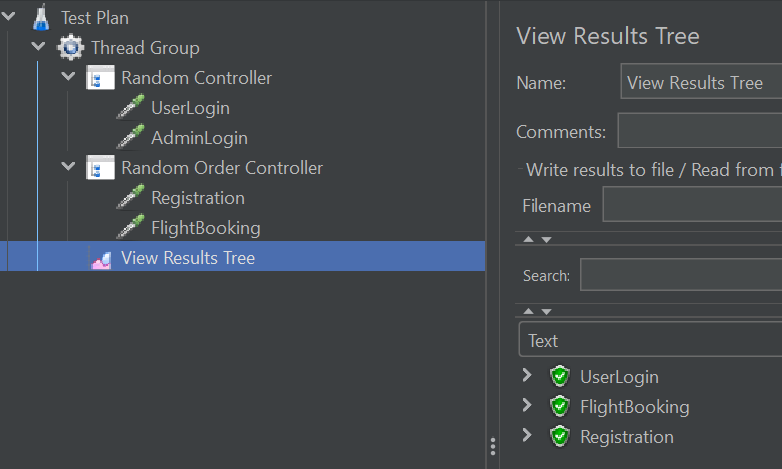
1. Random Controller –

It is the container where only one requests is taken randomly.

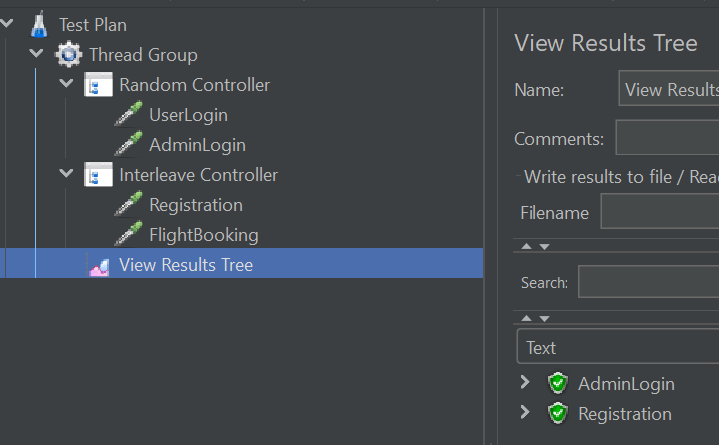


1. Random Order Controller –

It is the container where requests are taken in a non – sequential order.



1. Interleave Controller –



It is the container requests are taken in a sequential order. If thread count is only one, then it takes first one. If thread count is two, then it follows sequential order.

1. Modular Controller – To run a particular request multiple times whenever we need.

Steps:

* In the below picture, all the requests run in a sequential order.

Graphical user interface, text, application

Description automatically generated

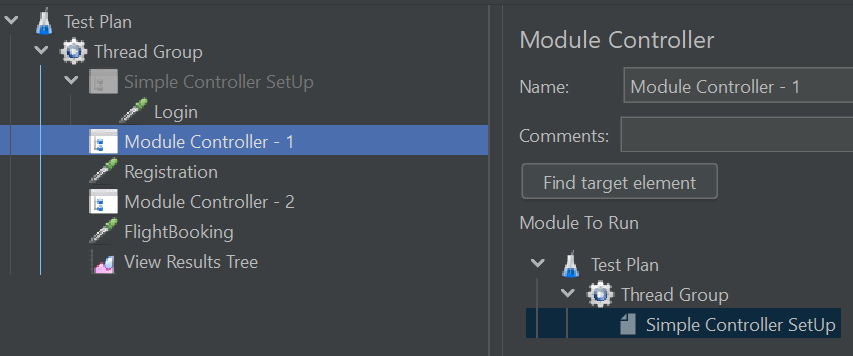
* Now, create a simple controller request and add ‘Login’ into it.

Graphical user interface

Description automatically generated with medium confidence

This also remains same as step – 1 if you run it.

* Now, add two ‘Module Controllers’ and add ‘Simple Controller Setup’ under each module controller. And disable the ‘Simple Controller’.



* Now run the requests and see the results.

A screenshot of a computer

Description automatically generated with medium confidence

1. Test Fragment – It is same as the Simple Controller.

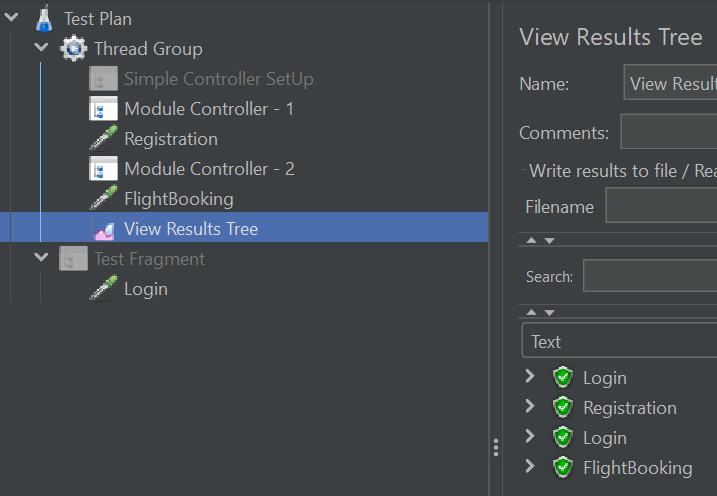
Steps:

* Create one ‘Test Fragment’ and drag ‘Login’ request and drop into it.
* And select the ‘Test Fragment’ in both module controllers.

Graphical user interface, application

Description automatically generated

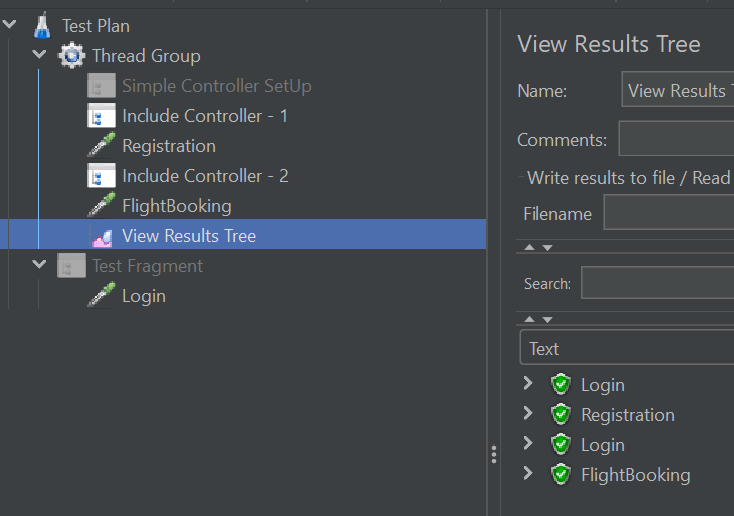
* Now run the ‘Test Plan’.



1. Include Controller – It is also same as module controller but here need to add ‘Test Fragment’ file.

Steps:

* Click ‘Save as Test Fragment’ for the ‘Login’ request under ‘Test Fragment’.
* Create two include controller and browse and add a file and run it.



1. Throughout Controller – Throughout controller provides the percentage of load to the request.

Graphical user interface, text

Description automatically generated with medium confidence

1. Recording Controller – Recording controller saves the action of request.

Steps:

* Add Non-Test Element – http recording.
* Set proxy on browser.
* Start recording in different recording controller.
* Start scenario recording.
* Stop recording and save.

1. Overview on performance testing –

* What is performance testing –
* Performance testing is defined as a type of ‘Non – functional’ testing to ensure software 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 performance testing is mainly on –
* Speed
* Scalability
* Stability
* Performance testing is also called as ‘Perf Testing’.
* Why do performance testing –

To test the application speed before going to release the product in market.

* Types of performance testing –
* Load testing
* Stress testing
* Endurance testing
* Spike testing
* Volume testing
* Some common performance problems –
* Poor scalability
* Poor response time
* CPU utilization
* Memory utilization
* Network utilization
* Operating system limitations
* Performance testing process –
* Identifying test environment
* Plan and design tests
* Identifying performance acceptance criteria
* Configure test environment.
* Implement test design.
* Execute tests.
* Analyse and reports.
* Performance Test Tools –
* Apache JMeter
* Micro Focus Load Runner.
* Neo Load
* Smart Bear ‘Load Ninja’.