**JMeter Performance Testing**

**Topic 1: What is JMeter?**

It’s basically performance testing tool, Java open-source software; supports multiple platforms, we can conduct multiple number of testing: load testing, stress testing, volume testing; it may also be used for testing database servers and API testing.

**Advantages:**

**- open-source license** (it’s totally free, just download and use it)

**- friendly GUI** (no need much time to adjust to the interface and learn deeply each component)

**- platform independent** (it’s purely Java desktop app, so it supports Linux, Mac, Windows and so on)

**- full multi-threading framework** (allows concurrent task execution)

**- visualize test result** (results might be displayed in different format: charts, diagrams, tables, log files)

**- easy installation** (just run small .bat file to launch JMeter)

**- highly extensible** (supports different visualization, plugins, testing strategies)

**- simulation** (can simulate multiple users in concurrent threads)

**- supports multiple protocols** (not only web application, but also server testing, API testing…)

Let’s see how JMeter works:

Изображение выглядит как текст, снимок экрана, диаграмма, дизайн

Автоматически созданное описание

---------------------------------------

**Topic 2: How to install JMeter on Windows?**

Pre-requisite is to have java in your system! Then go directly to the website jmeter.apache.org and download corresponding zip archive. Afterwards unzip your archive and replace resulted folder to C drive. Then open it, reveal bin folder and launch jmeter.bat if you wish to on admin’s behalf. That’ll start JMeter desktop UI.

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Значок на компьютере

Автоматически созданное описание

----------------------------------------------  
  
  
**Topic 3: JMeter Elements:**

There are different components of JMeter called elements. Each element is designed for a specific purpose.

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

At first let us concentrate on 4 components:

* Thread Group
* Samplers
* Listeners
* Configurations

**1. Thread Group**

It is a collection of threads, where one thread represents one user employing the application under test. Just imagine that every thread generates one real user request to the server.

We’re allowed to set up the number of thread (users) for each thread group.

For example: if I set 100 threads for one group, thereby JMeter will create and simulate 100 user requests to the server under test.

**2. Samplers**

They are different type of requests sent by Thread group.

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

**3. Listeners**

They show the results of test execution. They may provide results in different formats such as tree, table, graph, or log file.

Graph result listeners display the server response times on a Graph.

View Result Tree shows 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 result in the text file.

**4. Configuration Elements:**

Set up defaults and variables for later use by samplers.  
These are commonly used configuration elements in JMeter:

\* CSV Data Set Config

\* HTTP Cookie Manager

\* Login Config Element

\* HTTP Request Defaults

\* FTP Request Defaults.

---------------------------------------

**Topic 4: First JMeter Test Creation**

First of all, we gotta start JMeter, by launching jmeter.bat file and create Test Plan (but fortunately it’s already created by default). And I start developing Thread Group (called Users)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Actually I leave all the configures untouched (default settings), where we might notice following parameters: number of threads equals to 1, ramp-up period in seconds equals to 1, loop count equals to 1;   
  
Then within my Users Thread Group I create some elements (exactly one sampler with HTTP get request, and two listeners for displaying the results in format of tree and table)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Then I run my thread group and after it I’ll be able to observe my results:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

On the top line of interface there are different options we operate with. Run, Start, Stop, Shut Down, Save, Help…

---------------------------------  
  
  
  
**Topic 5: Timers**

What are timers?   
JMeter sends requests without applying any delay between each sampler. And if we conduct load/stress testing and try to gain realistic results from testing we must provide some breaks to prevent overloading situation on a server.

So JMeter timer is a solution to toss little pauses before sending every next request.

There are many timers available: Constant timer, Uniform random timer, Gaussian random timer, BeanShell timer, BSF timer, JSR223 timer.

But we’re gonna mainly focus on Constant timer and Uniform random timer. These are most popular.

Opened my application and for the prior settings I created trivial thread group with one thread (user ) 1 – ramp-up period and 1 loop. Added 3 same samplers (HTTP requests) with the same address:

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание

Then I pushed Constant timer to the Thread Group (it means I created it on the Thread Group Level which is not the only option) and I set up the gap equals to 5000ms = 5sec between the requests:

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание  
  
And I got these results in the table format:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Here’s another example where I specify my second timer at the Sampler level, I assign Constant TimerA to the first request with the same waiting time as Constant Timer B determined at the Thread Group level.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание  
  
That means that before executing the first request there will be 10-second break (Constant TimerA + Constant TimerB), but before following requests (second and third) we’re only expecting 5-second pause, because Constant TimerA belongs only to first request.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Now let’s consider Uniform Random Timer.

Sometimes we have to pass random time as a delay before executing the requests.

Now observe the interface of such Timer:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

As we can see we can configure random delay Maximum and Constant Delay Offset, and the time will be calculated based on this formula:

Изображение выглядит как текст, снимок экрана, Шрифт, чек

Автоматически созданное описание

I decided to preset timer’s configurations this way (And I also increased the number of user(requests) to three), and notice my timer is disposed on the Thread Group level that meant It’ll touch all the samplers:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Now we can analyze our results:

Изображение выглядит как снимок экрана, текст, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

-----------------------------  
  
  
  
  
  
**Topic 6: Controllers in JMeter | Loop Controller**

Logic Controller let you handle the order of processing Samplers/Requests in a Thread.   
Logic Controllers will decide “When&How” to send a request to a web server

JMeter provides multiple number of Logic Controllers, which are as follows:

* Critical Section Controller
* ForEach Controller
* If Controller
* Include Controller
* Loop Controller
* Module Controller
* Random Controller
* And others…

Today we’ll consider one particular controller which is called Lop Controller:  
it makes the user request run a specified number of times or run forever.

Изображение выглядит как текст, Шрифт, диаграмма, снимок экрана

Автоматически созданное описание  
  
  
I launched JMeter application and conventionally created and set up preliminary configurations, observe this (the only unusual thing here is number of loop count equaling to 2):

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

But supposedly I’d like to run some specific particular request 10 times (not 2) for that I can hang up one loop controller for a single request/sampler.  
  
Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

All I’ve done here was simply added loop controller to the thread group and then put my first request to this controller, so I made this sampler child of loop controller, and as you may notice I change number of loop counting to 5, that means that my first requests should be executed 10 times (5\*2), where 2 is a loop counting number for the whole thread group.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

And we may play with configurations by our way, thus I can dispose my resulting tree within loop control element as well to watch deliverables concerning only that particular tier. Observe this:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

-----------------------------

**Topic 7: Recording in JMeter**

We can record our actions on website and see the results in JMeter. For that we need to complete few steps pf algorithm:

1) Add NonTestElement --- http(s) test script Recorder

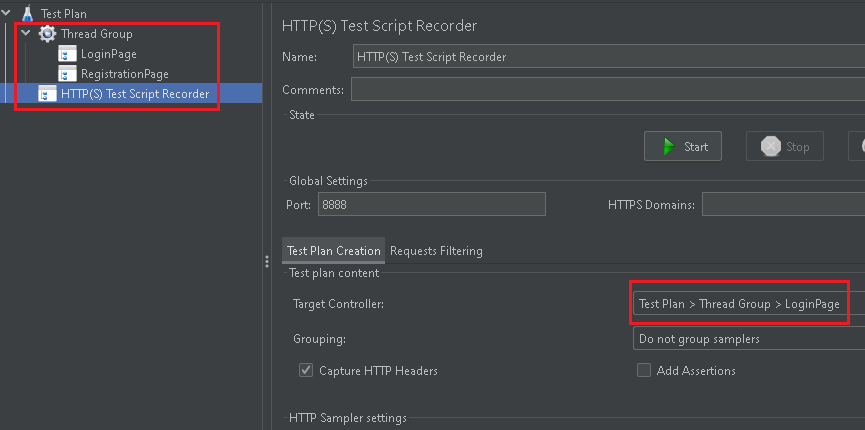
2) Set Proxy on Browser

3) Start Recording in different Recording Controllers

4) Start Scenario Recording

5) Stop Recording and Save

Alright, let’s proceed to the task. At first I’ll dig into configuration part and set up a couple properties within JMeter application.



On the screen we may observe following setups: I added nonTestElement where I’ll later start the recording process from. Port number is predefined by default as 8888. My first referring controller is LoginPage (Recording controller created on ThreadGroup level to write all the notes directly in there)

After completion this part I’m moving on and altering the proxy settings like it’s presented on the picture below:

Изображение выглядит как текст, снимок экрана, Шрифт, число

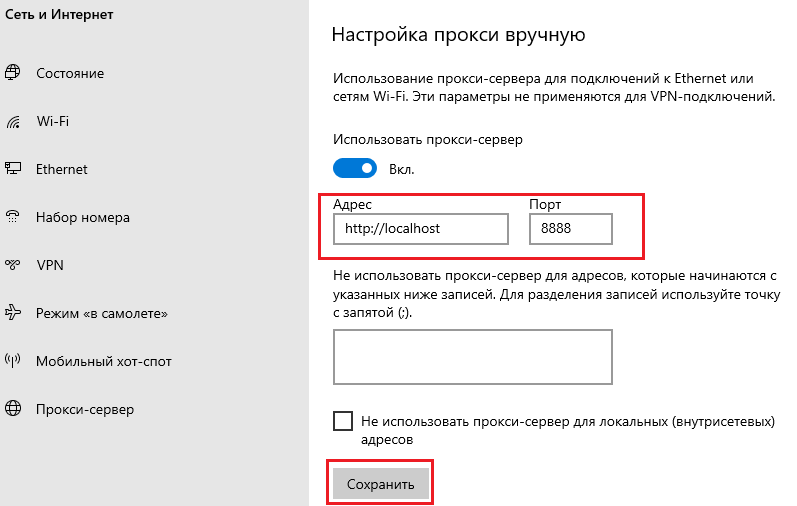
Автоматически созданное описание

Then immediately I upload imperative certificate to the firefox browser (in my case):

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

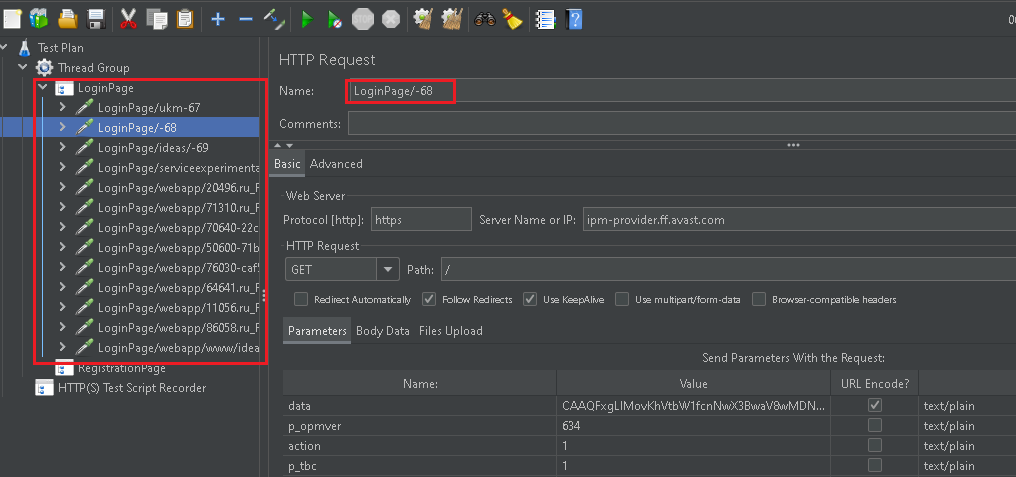
Автоматически созданное описание

Finally I visit my personal settings on the PC (proxy section) and adjust it this way:



Thus, I have established basic configurations for recording actions and displaying them onto JMeter system.

Next step is launching the record controller. So I need to click on ‘Start’ button and It’ll start recording. Later I open my browser and surf through different sites by clicking on links. As a result I’ll get all the uploaded resources from those sites inside my LogInPage controller:



And of course we have to set back the proxy related configurations as long as we don’t make any recording activities.

Ultimately, to retrieve the expanded outcome I can add one listener to get all the preliminary results displayed in the tree format or whichever we wish:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

----------------------------------

**Topic 8: Disparate Controllers in JMeter**

* Simple Controller
* Modular Controller
* Test Fragment
* Include Controller

Simple Controller is usually used to contain multiple requests in it. For grouping several requests.

At first I got defined with resource I’m gonna adhere to and it is some flightbooking website, here it is:

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Then I composed a simple structure within JMeter we’ve been familiar with for a while. That’s how it currently looks:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

On the screenshot above there are created samplers representing http requests to various resources of the flight booking website, which btw I’ve already executed once to view the outcome.

By the next step of elaboration, I’m tuned in to add simple controller which will store all my samplers as a container or a storage.

Thus, I’m adding Simple Controller and displacing it onto Thread Group level:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Regardless I have added Login Sampler under the Simple Controller and not touched other Samplers I may observe the same picture I got last time.

Now I want to add ‘Module Controller’ to execute the requests in a certain order. Look at the screenshot below and then I’ll explain current configurations:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

First of all, I appended two more controllers which both relate to Module Controllers. I’ve put them before Registration and FlightBooking samplers in order to execute these requests in my specified way, module controllers 1 and 2 are exactly the same and both refer to Simple Controller that represents a container. By the way I disabled Simple Controller. Let’s run our build and see what’ll happen.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Yes, exactly what I expected to derive from this execution.

So far we have considered Simple and Module Controllers. There’s one option to use and this is not exactly a controller, but might perform as an alternative for Simple Controller because it has mostly the same functions and demeanor. Let’s observe. I arranged the same situation  
  
Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

I added Test Fragment inside my Thread Group where I put Login Sampler in. And now as you noticed at the picture above I refer not to simple controller but to test fragment. And I intend to receive the exact same result as at previous execution.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Yes, perfectly fine. It works as expected.

At the end of our session let me take a few minutes to mention last but not least controller for today. This is about ‘Include Controller’. Here’s a brief plan. I not only disabled but also removed previous controllers preliminary saved Login sampler as a test fragment (there’s option by right-clicking on a sampler). After I added two Include Controllers how It was with Module Controllers. And I just refer to a saved Login Test fragment through Include controllers to deduce them and make be executed.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Eventually we may enjoy the same results:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

---------------------------------------------

**Topic 9: Random Controller & Random Order Controller**

1) Random Controller is a special controller in JMeter which will pick one of considered samplers(requests) belonging to this particular Random Controller. For clarity we can assume that these samplers are like children for Random Controller and only one of them may be executed.

At first I arrange following settings, I created a Thread Group and put a Random Controller in it; later I assigned my samplers to this controller, I also added Listener for tracking the final results.Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

As we can notice, there’s only one request that’s been executed out of three. Because random controller accomplished its deed by choosing randomly from the offered roster.

Now, let’s move on and take a look at Random Order Controller. All the mutations I have to do in my prior settings is to change Random Controller on Random Order Controller.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

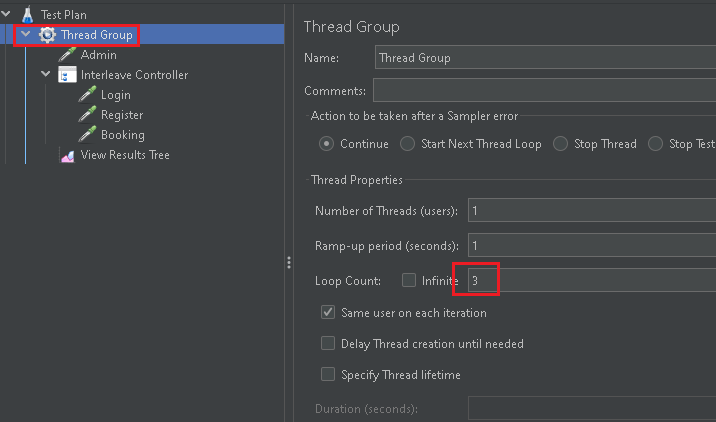
Автоматически созданное описание

Difference between Random Controller and Random Order Controller lies in number of requests which are considered being executed. In Random Order Controller each request gets executed but in random order as we may notice.

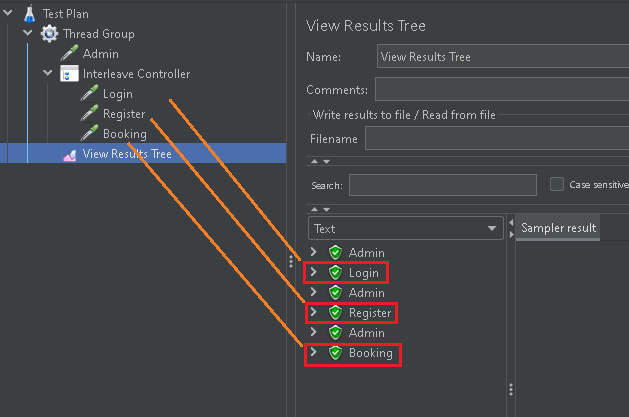
-----------------------------

**Topic 10: Interleave controller in JMeter.**

This is a peculiar controller used in this system when I want to send my request sequentially through several loops one by one. Here’s a visible/tangible example:



I have created this structure where I assigned ‘3’ to loop count variable for more perceptible result (since I put 3 samplers into Interleave controller group). Now let’s observe the final result after execution onto ‘View Result Tree’ listener:



There were 3 loops at this particular launch according to preliminary settings. Now you can guess the logic this program has been executed by.

-------------------------------

**Topic 11: Throughput controller.**

This controller exists to perform distributed load test, so by employing this controller I may emulate work where I sort of divide all users for certain number of groups.

Let’s imagine I own 10 users (threads) and I want 2 of them to make visit one site, 3 of them – another site and 5 of them – third site. I can implement 3 separate throughput controllers to provide this kind of service.

I have created the following structure of controllers with samplers in the view of http-requests under the thread group where I defined 10 threads (users):

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Now I need to set up configurations of throughput controllers and establish percent executions for these controllers (20% for 1st cntrl, 30% for 2nd cntrl, 50% for 3rd cntrl).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Then I ran my test plan and received the result into aggregate report (according to my prior settings the ratio of samplers has been divided correctly):

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

I can increase start number of users and percentage will be the same for each throughput controller.