# Jmeter Notes

Contents

[Jmeter Notes 1](#_Toc16502543)

[Walkthrough From the Start 1](#_Toc16502544)

[When running from CLI 3](#_Toc16502545)

[Reading Data from CSV File 5](#_Toc16502546)

[Functions and Variables 7](#_Toc16502547)

[User Parameters 7](#_Toc16502548)

[Passing Variables Between Requests 10](#_Toc16502549)

[Setting up a Realistic Performance Test – Pacing 12](#_Toc16502550)

[Timers – How to Add Think Time 12](#_Toc16502551)

[Scheduling and Sequencing 14](#_Toc16502552)

[Scheduling test to run for specific number of seconds 14](#_Toc16502553)

[Running test sequentially 14](#_Toc16502554)

[Sequentially Adding Thread Groups to Test Run 15](#_Toc16502555)

[Correlation with Regular Expression Extractor 16](#_Toc16502556)

[Templates 18](#_Toc16502557)

[In Summary 21](#_Toc16502558)

[Using Test Script Recorder 21](#_Toc16502559)

[How to Test File Upload 26](#_Toc16502560)

[How to Test File Download 27](#_Toc16502561)

[Debugging 29](#_Toc16502562)

[How to record a test with Blaze Meter 33](#_Toc16502563)

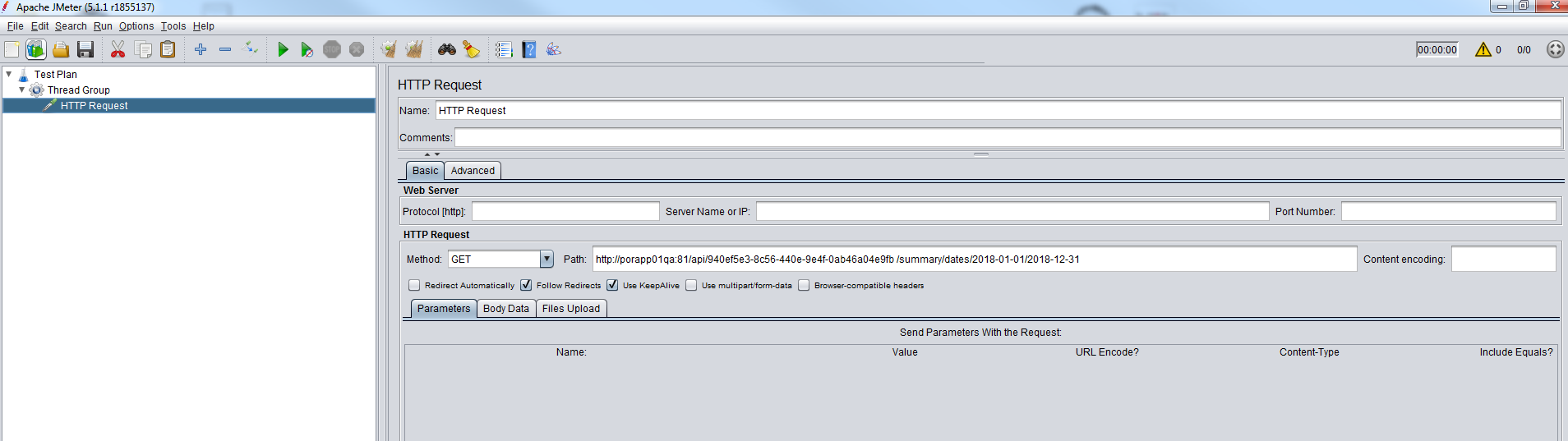
## Walkthrough From the Start

From home screen, looks like it is giving you a new Test Plan, so no need to select File->New

Right click on Test Plan and select Add -> Threads (Users) -> Thread Group

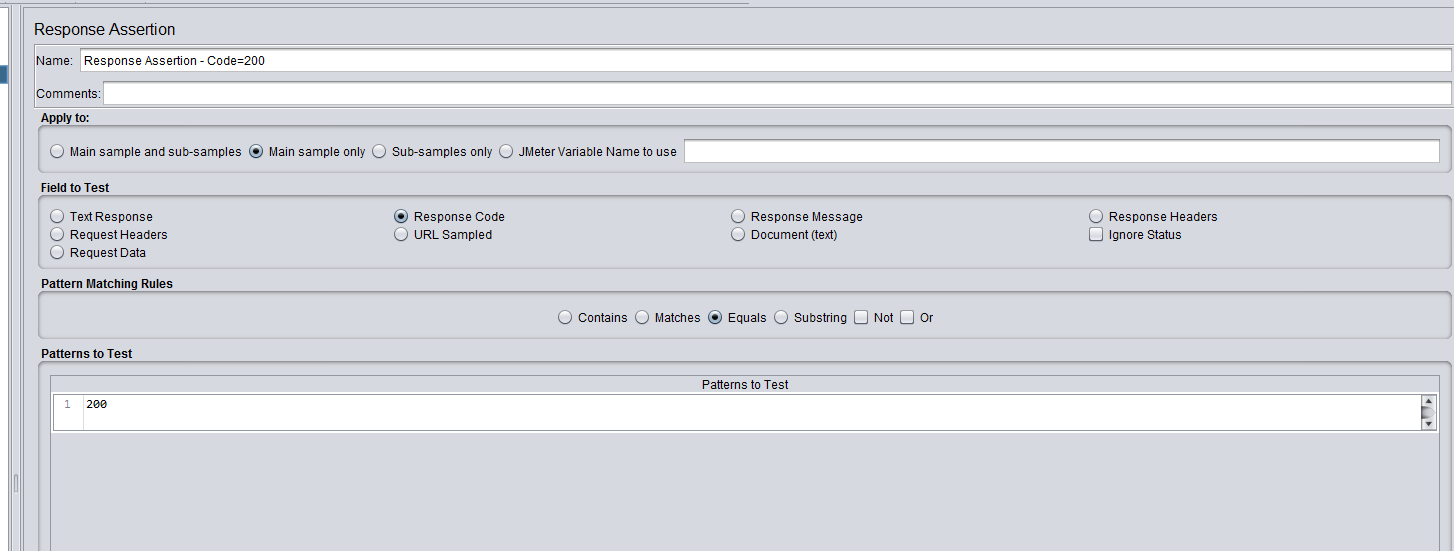
Right click on the Thread Group and select Add->Sampler-> HTTP Request

In the Path field, enter the URL for the request (same as Postman, but can’t use environment variables (may be able to parameterise this later?)



Right click on the HTTP Request and select Add->Assertions-> Response Assertion

In the Response Assertion, click on Add and enter the value (e.g. 200) in the Patterns to Test panel. Select Response Code in the Field to Test panel



Right click on the HTTP Request and select Add->Assertions-> Duration Assertion

Add a value to the Duration in milliseconds field

Right click on the HTTP Request and select Add->Listener-> Assertion Results

Right click on the HTTP Request and select Add->Listener-> View Results Tree

Right click on the HTTP Request and select Add->Listener-> View Results in Table

Select the Test Plan or Thread Group and click on the green arrow to run the request.

## When running from CLI

Format is

Jmeter –n –t {location of test plan file} –l {location of logfile} –e –o location of HTML folder for reports

For example (in this case we are already in the bin folder for JmeteR)

jmeter –n –t Basic1.jmx –l Log3.csv –e –o HTMLReportsX/

Points to note:

It does not seem to like file names or folder names with spaces and dashes (dashes are most likely to be the problem)

If the log file already exists, it will complain that it is not empty (is there not an option to append?)

If the HTMLReports folder already exists, it will complain that it is not empty (is there not an option to append?)

The folder name for HTML reports needs to have a slash on the end.

Although you can specify a new name for the log file and it will create the file for you, it will not do so for the HTML reports folder. The folder must already exist and be empty

Running like this seems rather hit and miss and exiting CLI and trying again seems to be the only solution.

The HTML Reports can also be created from an already existing log file instead of as part of a test run.

The format is

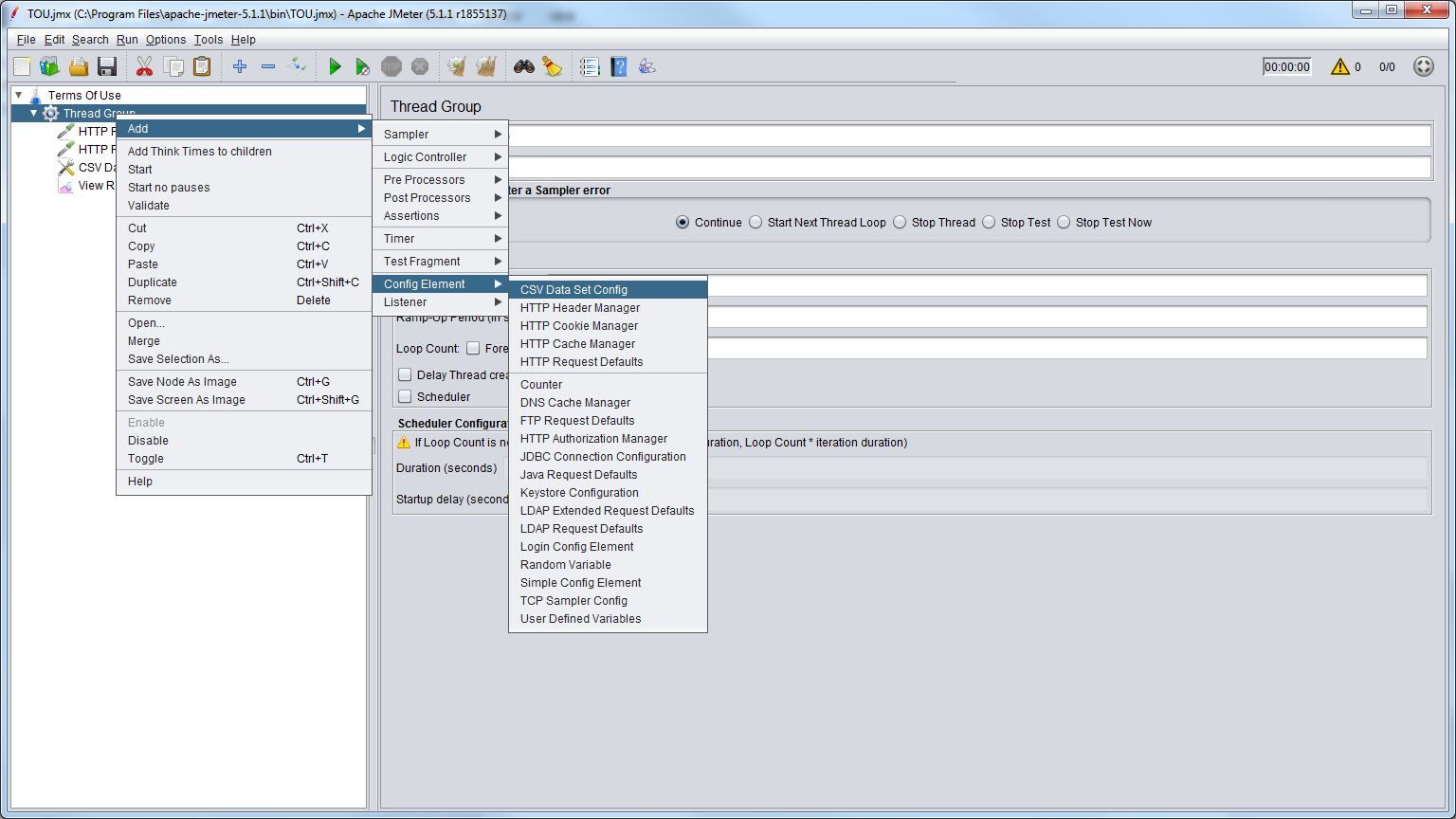
Jmeter –g {location of logfile} –o location of HTML folder for reports

For example (in this case we are already in the bin folder for JmeteR)

jmeter -g Log3.csv –e –o HTMLReportsX/

## Reading Data from CSV File

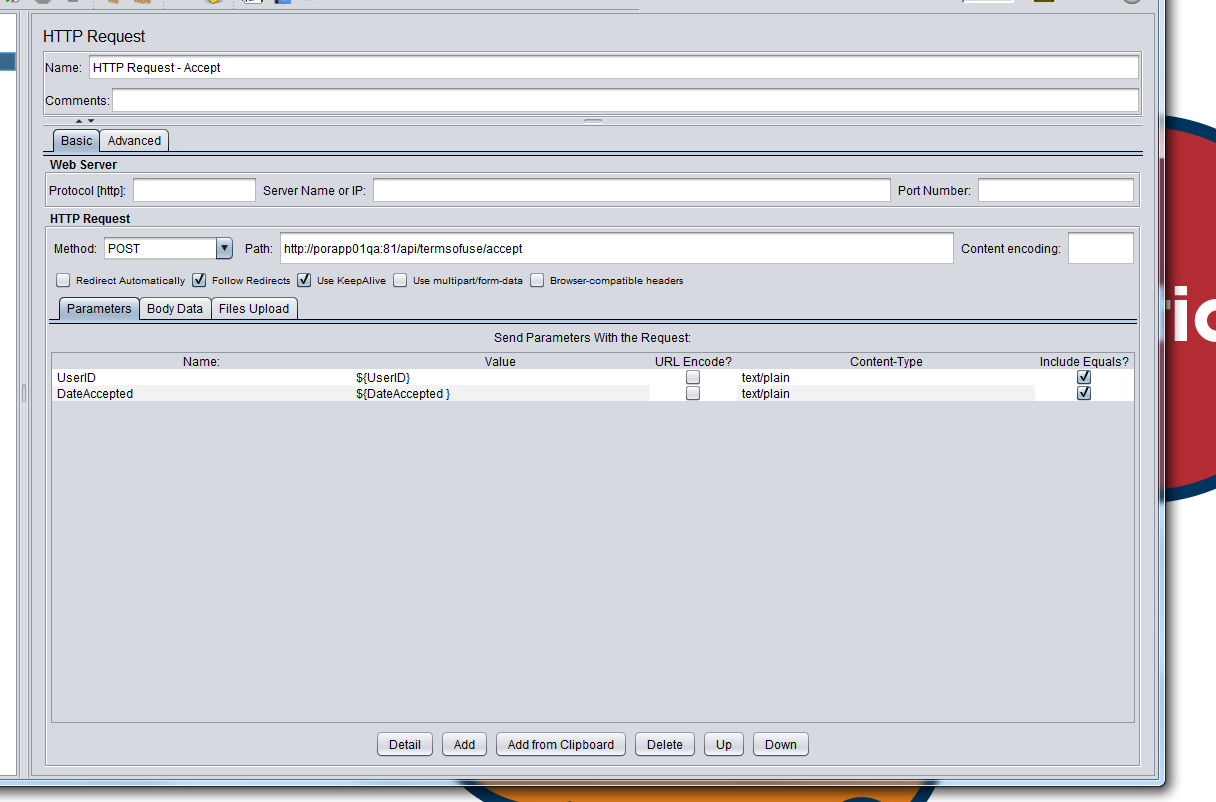
Right click on the Thread Group and select Add->Config Element -> CSV Data Set Config



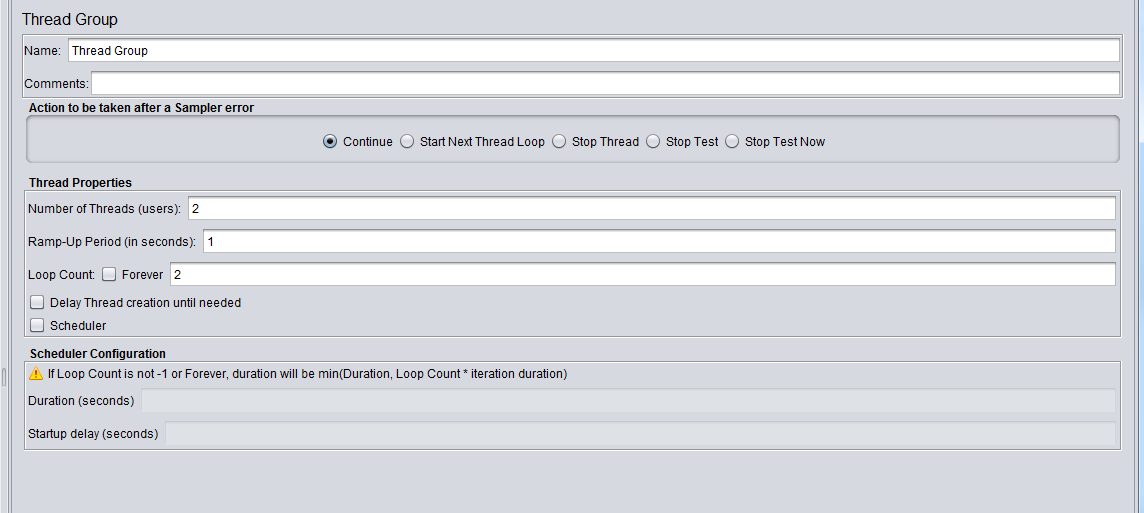
Set up your CSV data file with headings and data.

Enter the filepath and name (or just filename if it is in the JMeter bin folder) in the Filename field in JMeter.

In the Parameters tab of your HTTP request, replace the contents of the Value fields with the column headings of your file in the form ${columnName}



If you have multiple lines in your data file, but only 1 iteration in our look count, then it will only use the first line of input data, you will need to increase the number of iterations to use multiple lines (there is another way, see later)



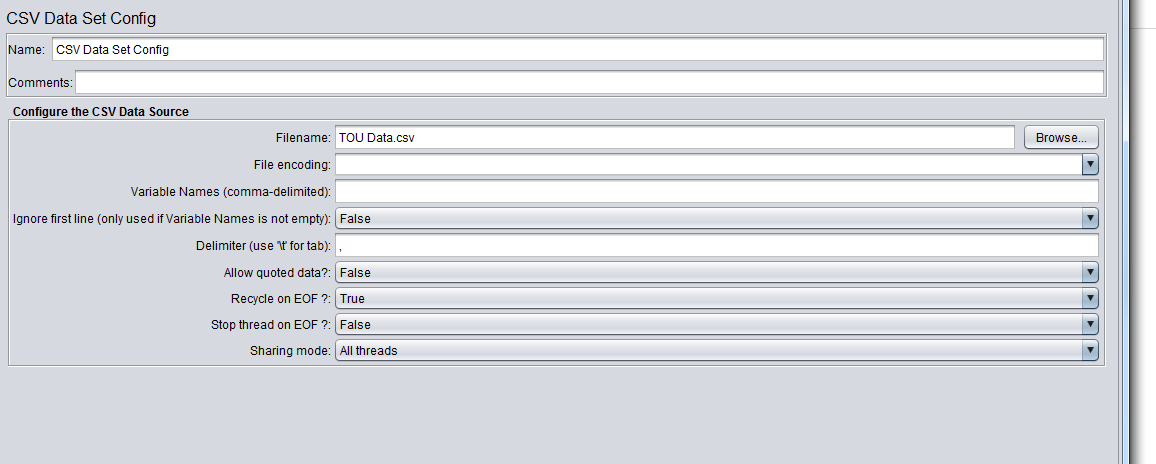
Instead of having column names, you can use the Variable Names field. Be careful not to have any column names if you use this approach, otherwise the column names will be taken as input data. Although it looks like you could get round this by setting Ignore first line to True.

Allow quoted data allows your input data to have quotation marks.

Recycle on EOF means that if you have more iterations that you have lines of data, then it will loop round to the start of the file and keep going until all iterations are complete.

Stop Thread on EOF means that if you have more iterations that you have lines of data, then it will stop at the end of the file, even though your number of iterations is not complete.

Sharing mode allows the input data file to be shared between threads rather than using one file per thread.



## Functions and Variables

Variables are always local to a thread.

Functions and variables are both case sensitive.

### User Parameters

To include things like a base\_URL or a user\_ID, you need to add User Parameters.

Right click on the Thread Group and select

Add-> Pre- Processors -> User Parameters

Enter the Name and value for the parameter

In the request, you access the variable using format ${variableName}, e.g. ${base\_URL}

(NOTE: the value goes in a column with name User\_1. It seems like this value will be taken for all users. You can add another user, probably if you want a different value for user 2)

There seem to be many built-in functions available, e.g.:

${\_\_log(“enter text to log here”)} – works a bit like writeln()

${\_\_time(dd MM YYYY HH mm ss)} – displays current date-time in given format

$(\_\_threadNum) – takes no arguments, displays the thread number. Of use when you have multiple users (threads).

${\_\_intSum(2,3, result)} – performs a sum (addition) on the given arguments and put it in the result variable.

This variable can be accessed elsewhere in the same thread by

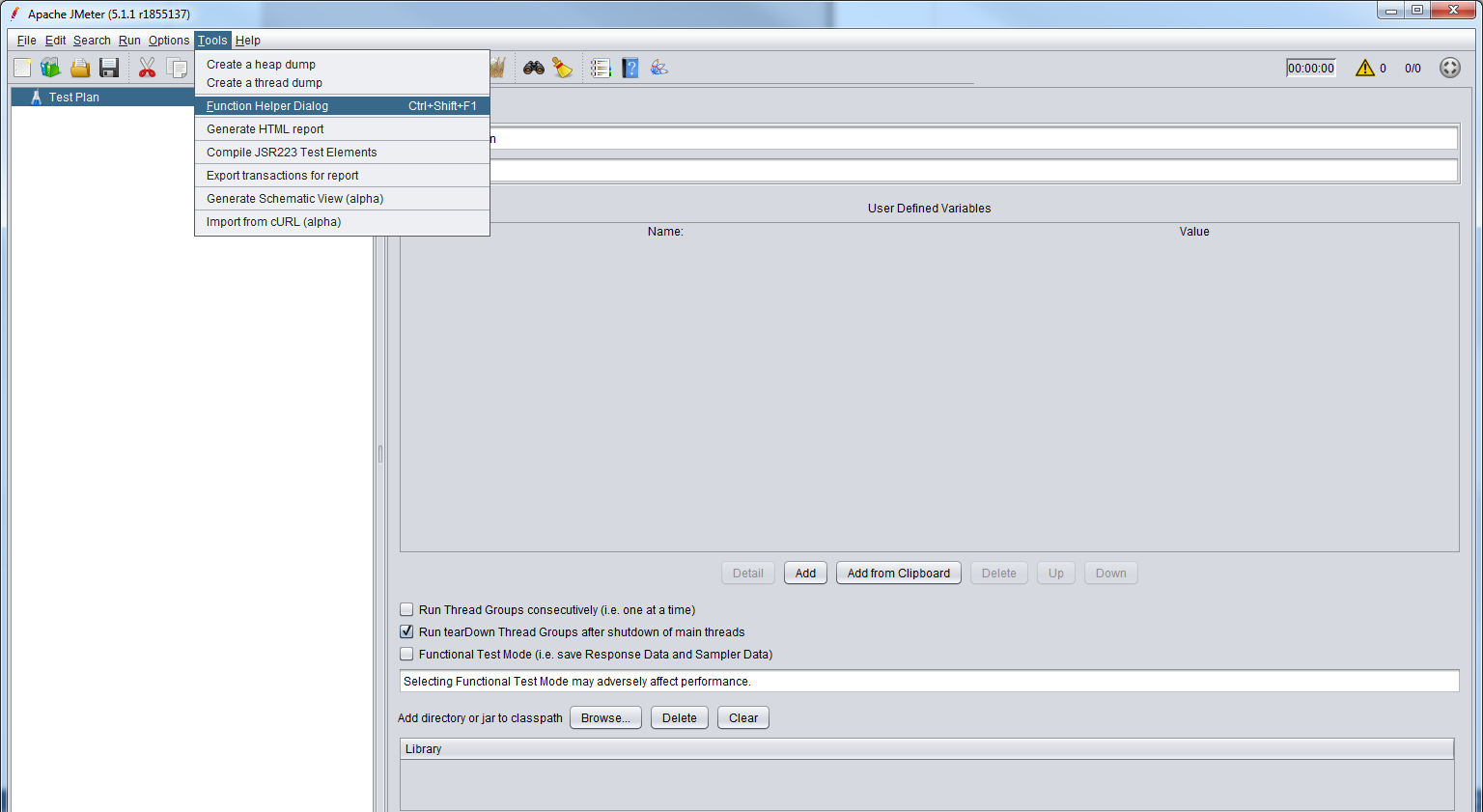
${result}

You can combine both variables and functions

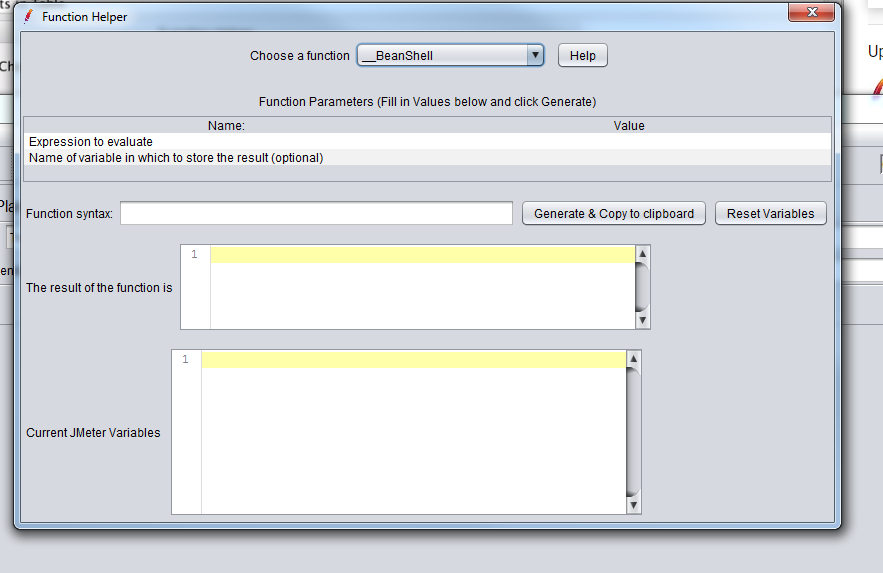
E.G.

${\_\_intSum(2, ${result})}

Get a list of functions, go to Tools -> Function Helper Dialog



Then you can see a list of all available function under Choose a function



You can then click on the Help button to get information on the selected function.

# Passing Variables Between Requests

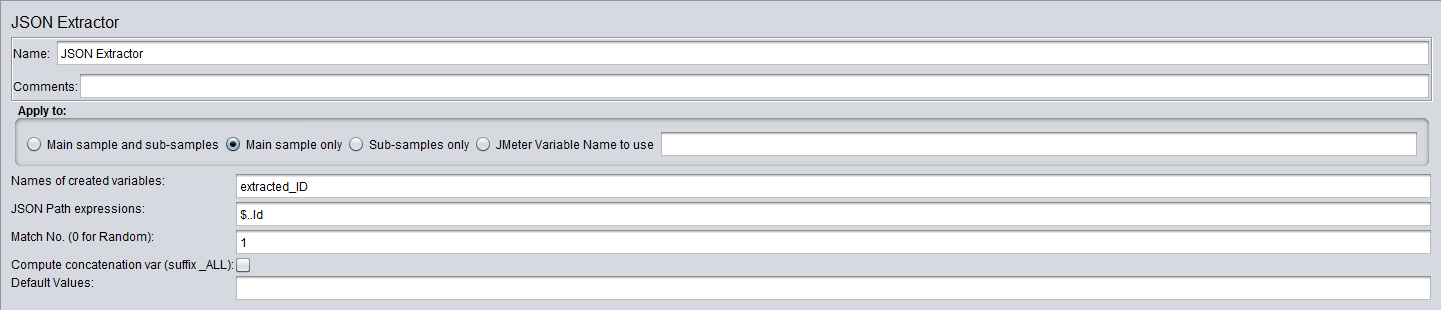
In order to get a variable from the first request in order to pass it to the second, you will need to use a JSON extractor:

From the Thread Group, right click and select Add-> Post Processors -> JSON Extractor

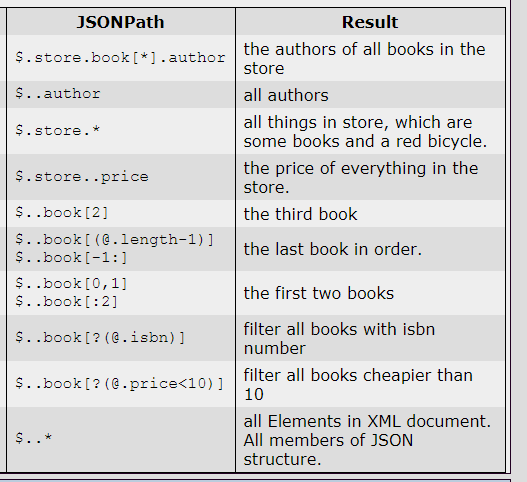
Drag this to just after the HTTP Request whose response you want to extract the value from

Enter a variable name to extract the value to and a JSON Path expression to find the value

E.G.

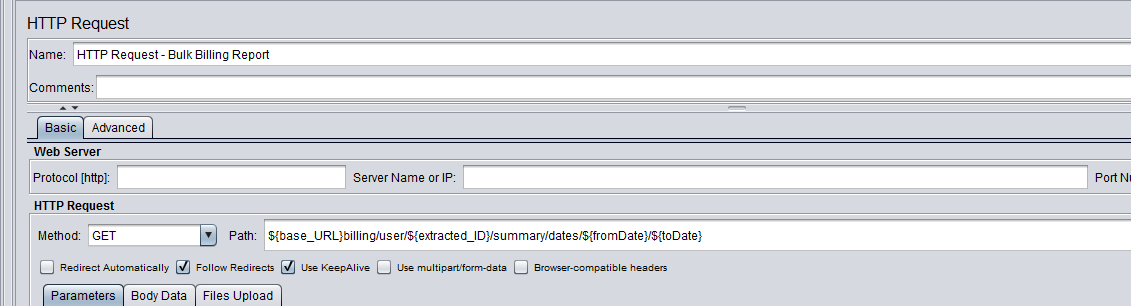


JSON Path Examples



In the Path of the HTTP Request that uses the extracted variable, enter it in the form ${variableName}

E.G.



After the HTTP Request and any assertions, add a Debug PostProcessor, this will allow you to see the result of the extraction to help with any debugging.

Right click on the Thread Group and select Add-> Post Processors -> Debug Post Processor

Drag this to correct position

## Setting up a Realistic Performance Test – Pacing

Need to add plug in called Stepping Thread Group.

Options -> Plugins Manager

Need to get round the proxy with something like this

jmeter -E https -H my.proxy.server -P 8000 -u username -a password -N localhost

See <http://jmeter.apache.org/usermanual/get-started.html#proxy_server>

Once done, you can add a Stepping Thread Group instead of a normal Thread Group

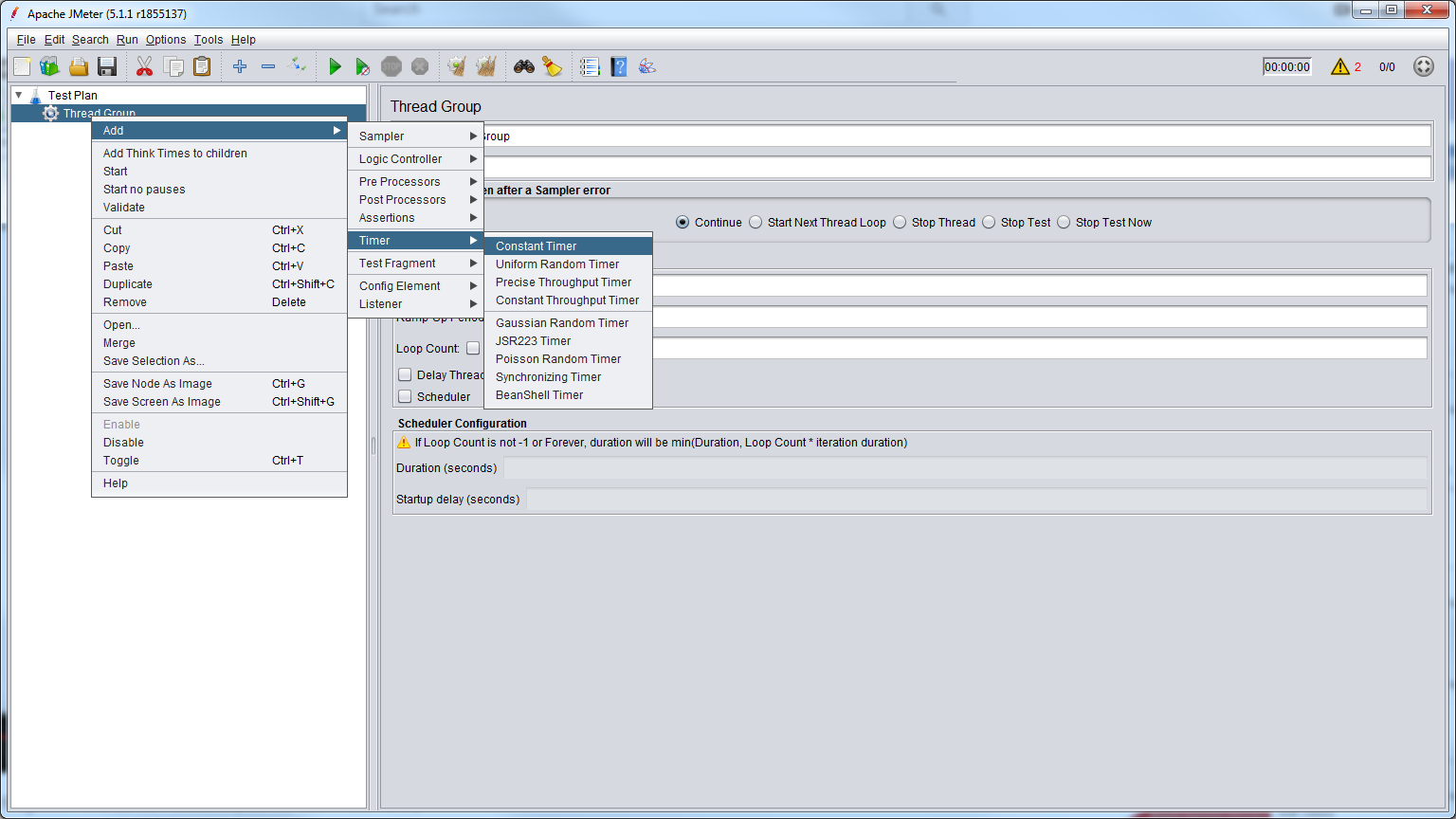
This has been deprecated, instead use Concurrency Thread Group. It is part of the same plug in.

See here for details of how to use.

<https://www.redline13.com/blog/2018/05/guide-jmeter-thread-groups/>

## Timers – How to Add Think Time

Adds delays, used to simulate the user’s think time



Constant Timer (fixed amount of time for delay) and

Uniform Random Timer (random amount of time for delay)

Should cover most situations.

Timers are local to the Request, Simple Controller or Thread Group in which they are placed.

So a Request could be affected by multiple timers depending on where they are placed.

Uniform Random Timer

The delay is calculated as:

0.x\*Random Delay Max + Constant Delay Offset

Where x is any value from 0 to 9

E.G.

Random Delay Max = 100

Constant Delay Offset = 1

Gives

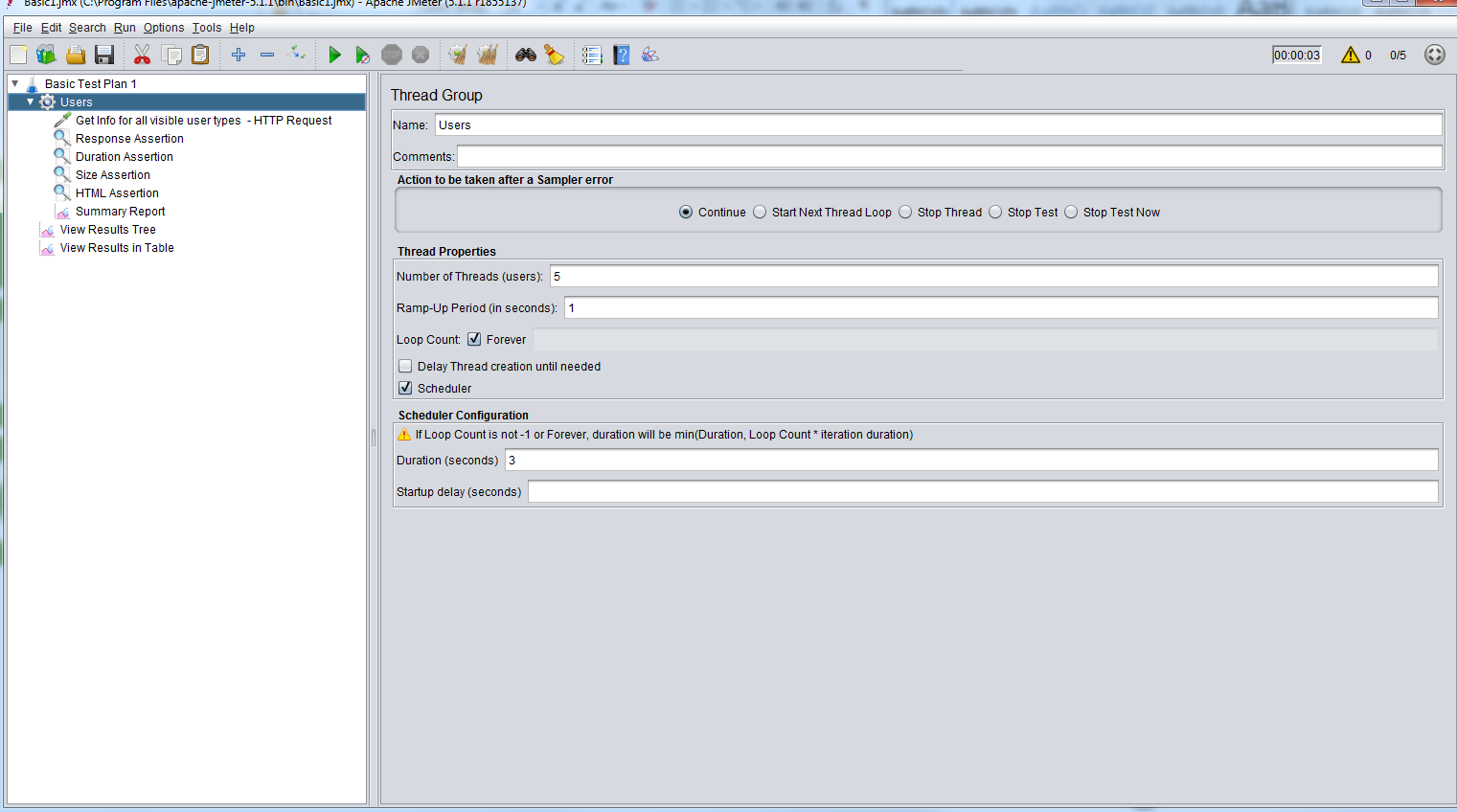
0-9 \* 100 + 1

= delay of 1 to 100 milliseconds

## Scheduling and Sequencing

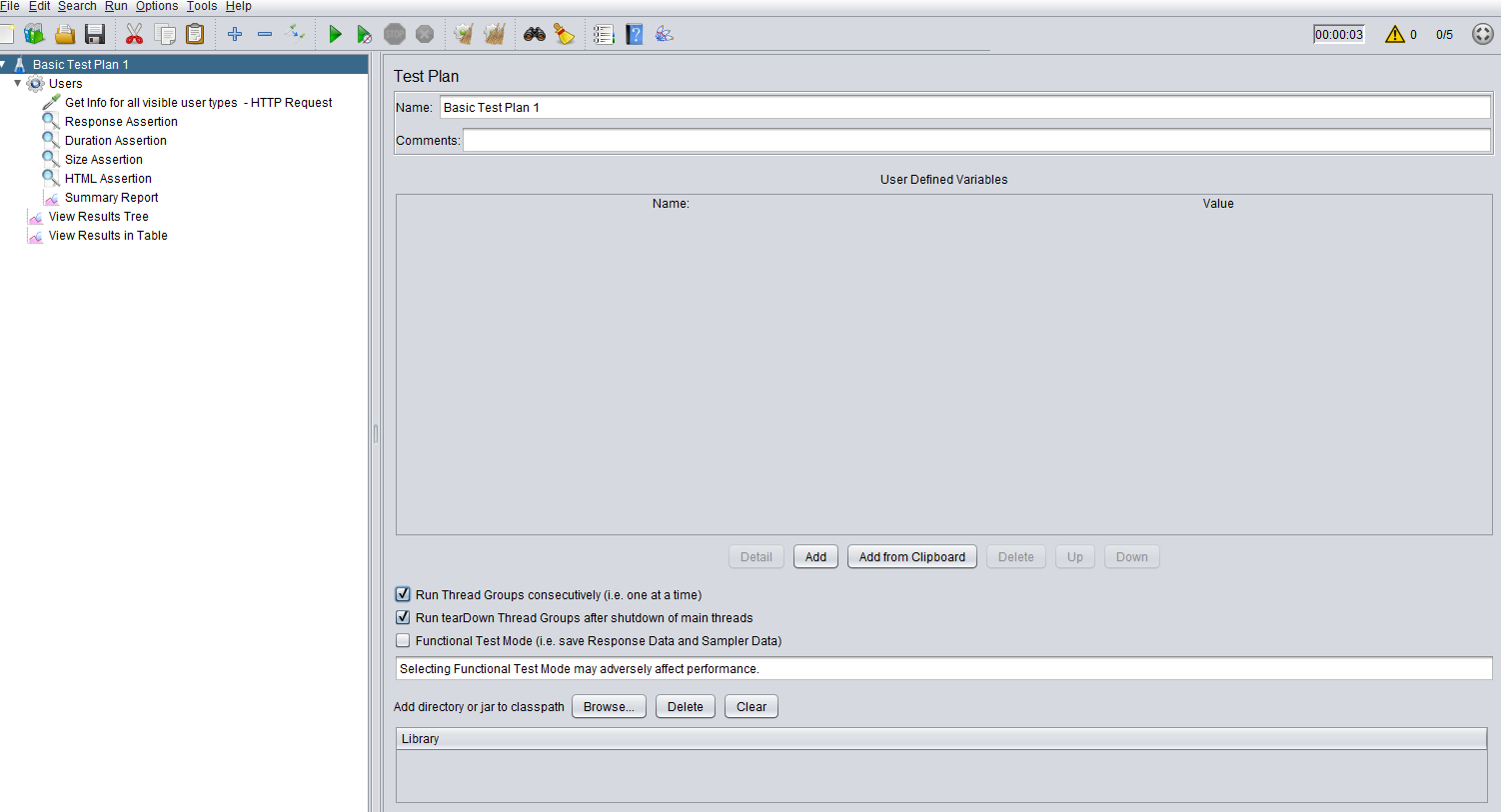
### Scheduling test to run for specific number of seconds

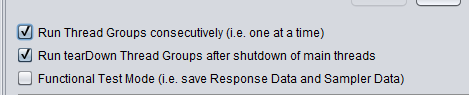
In your Thread Group, set Loop Count = Forever. Click on Scheduler tick box and set the duration in seconds.



### Running test sequentially

Got to Test Plan and select Run Thread Groups consecutively





So if you set threads 1 and 2 to run for 10 seconds each, thread 2 would only run after thread 1 had finished running for 10 seconds.

### Sequentially Adding Thread Groups to Test Run

If you want to run Thread Group 1 then to add Thread Group 2 while the first is till running:

Add duration (total) to first thread group

Add duration and delay to second group such that total = delay + second duration

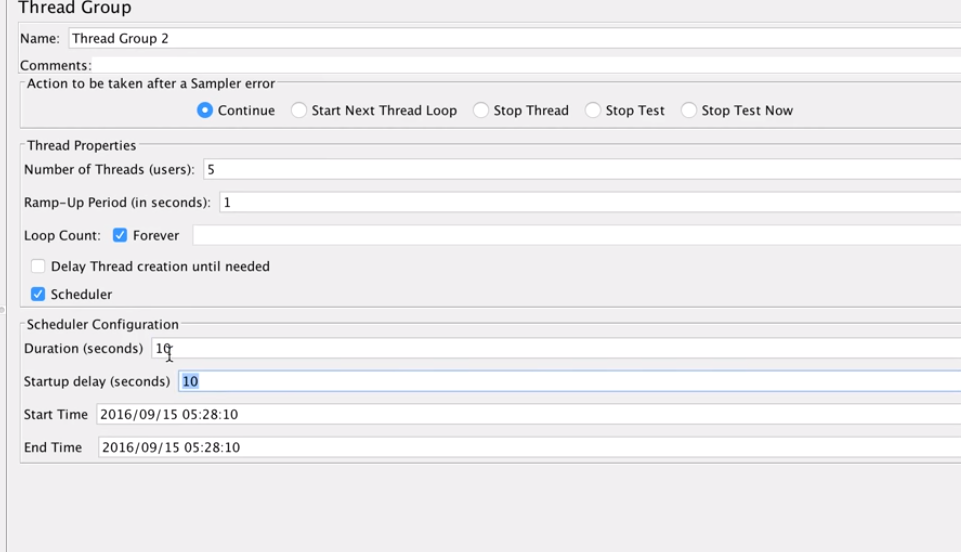
E.G.

Total Duration – 25

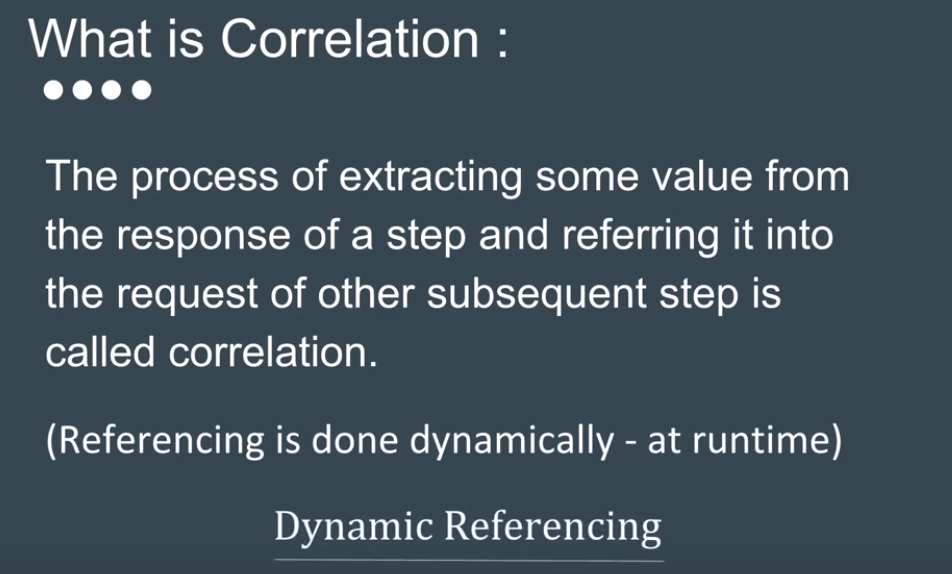
Delay = 10

Second duration = 15

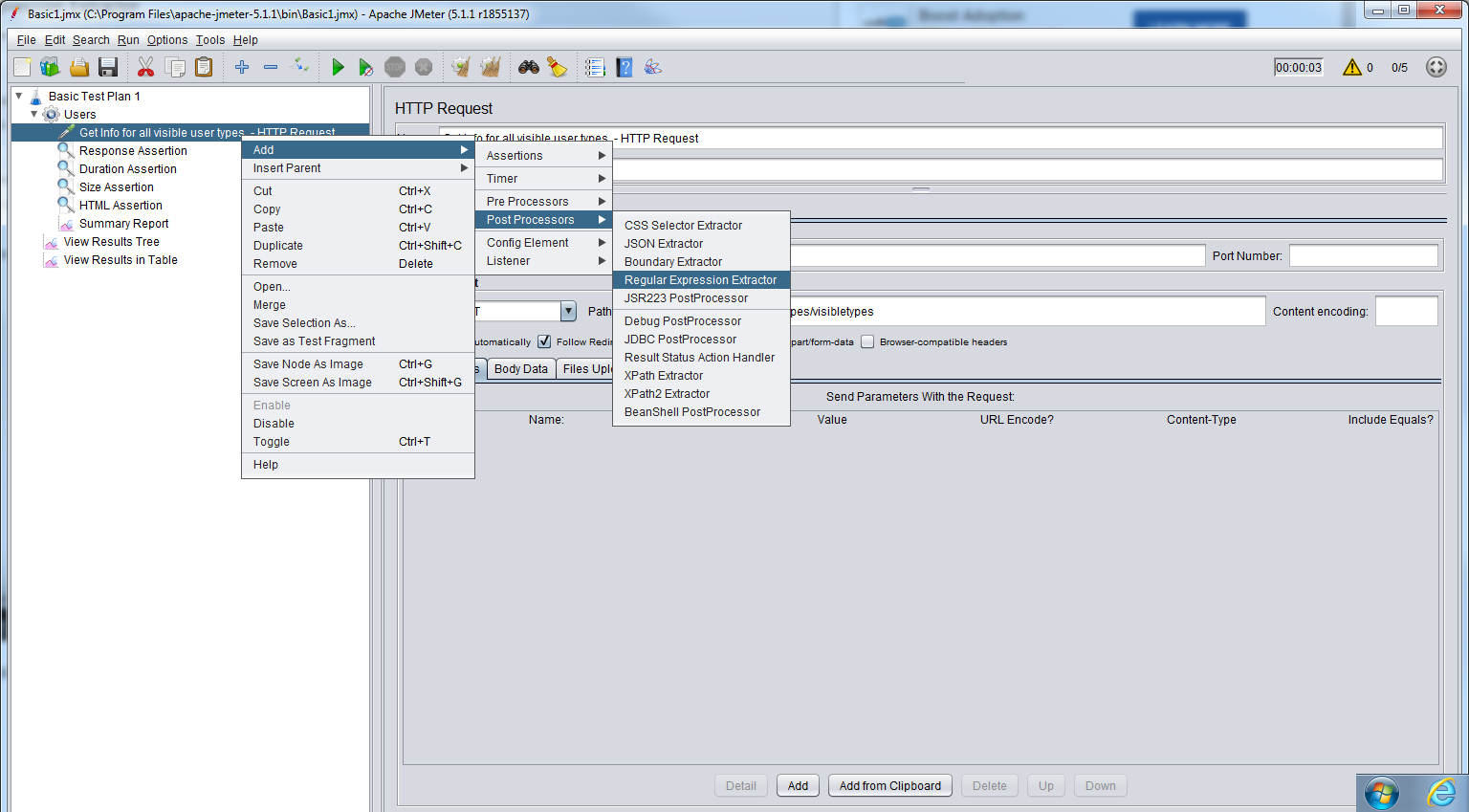
Thread Group 1 will run for 25 seconds and Thread Group 2 will run also run for the last 15 of these seconds



## Correlation with Regular Expression Extractor



From your request, select Add->Post Processors -> Regular Expression Extractor



In the Regular Extractor screen, you can select where the extraction will apply to (e.g. Main Sample) and what field to check (e.g. Body).

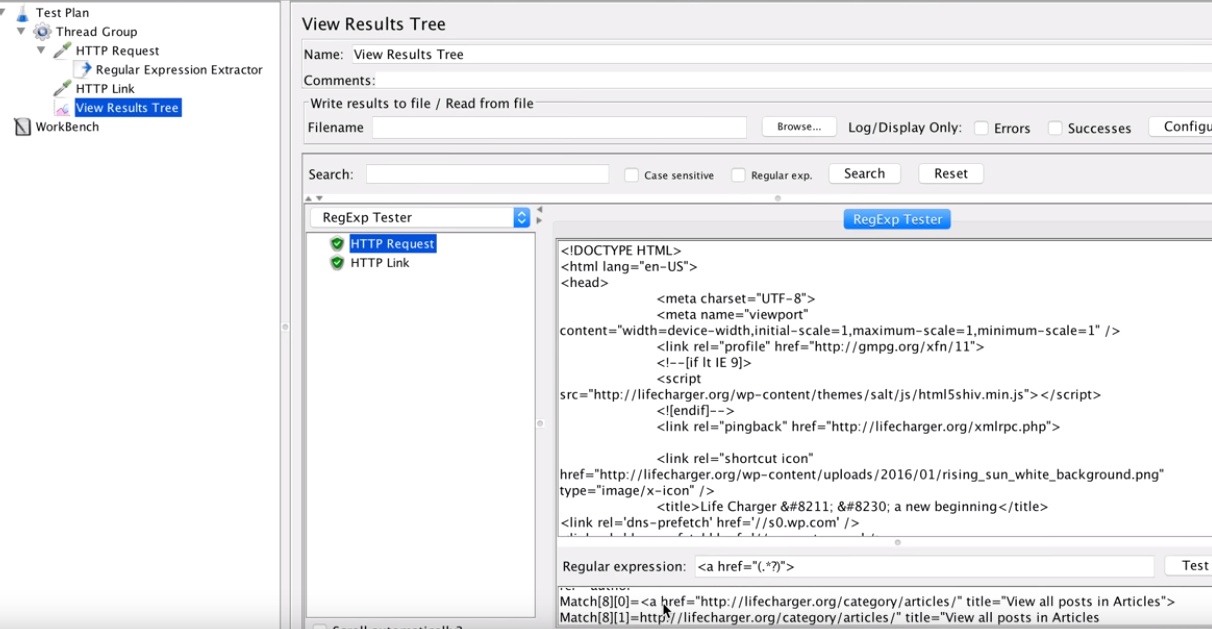
The reference name is the variable name where you will extract the variable to.

Regular Expression is the RegEx you will use to extract the required value.

Go to Regexr.com to help you create and test your regular expressions.

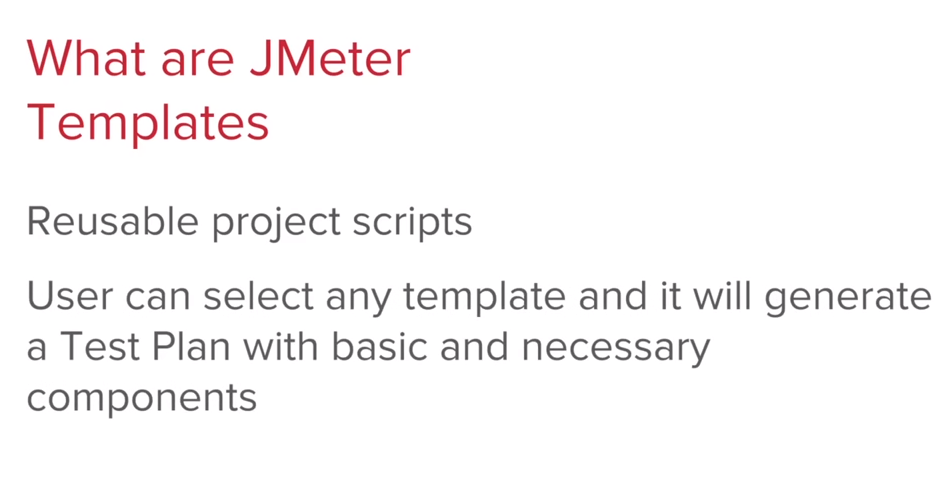
You can copy the output from your response and paste it in here and then create a regEx that finds the data you need.

You can also use the RegExp Tester in the View Results Tree in JMeter



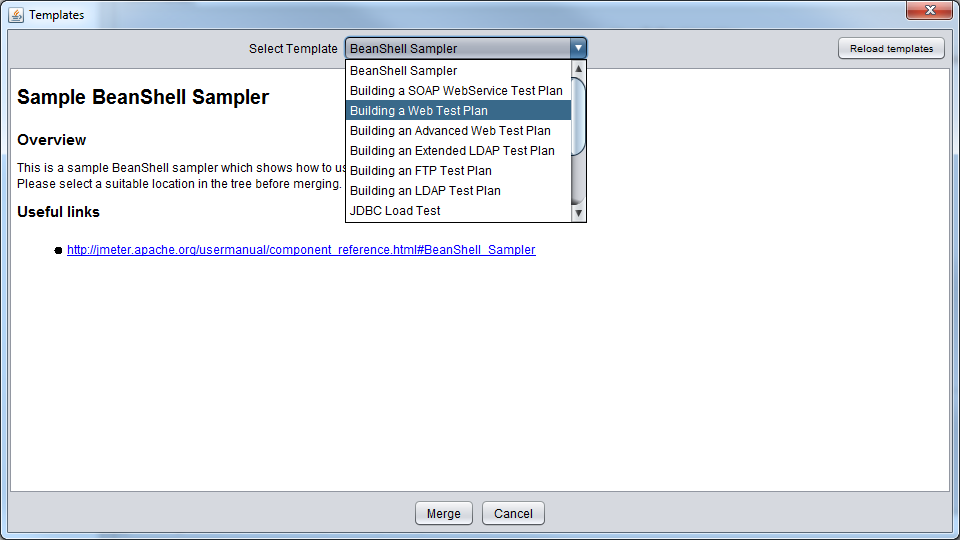
Once you have extracted the data into your reference you can reference it in other requests as you do with other variables $(ReferenceValue)

## Templates



To use a template, select File-> Templates.

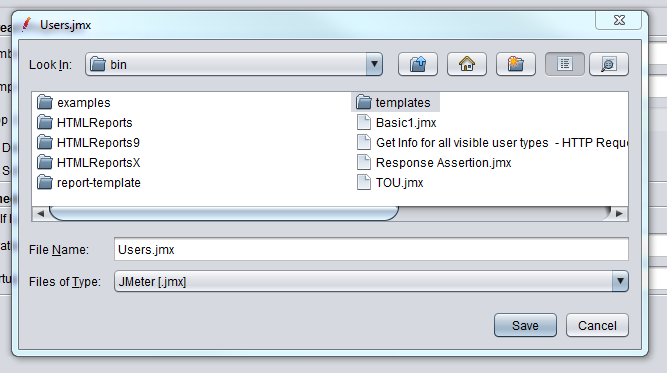
This will display a screen where you can select a template:



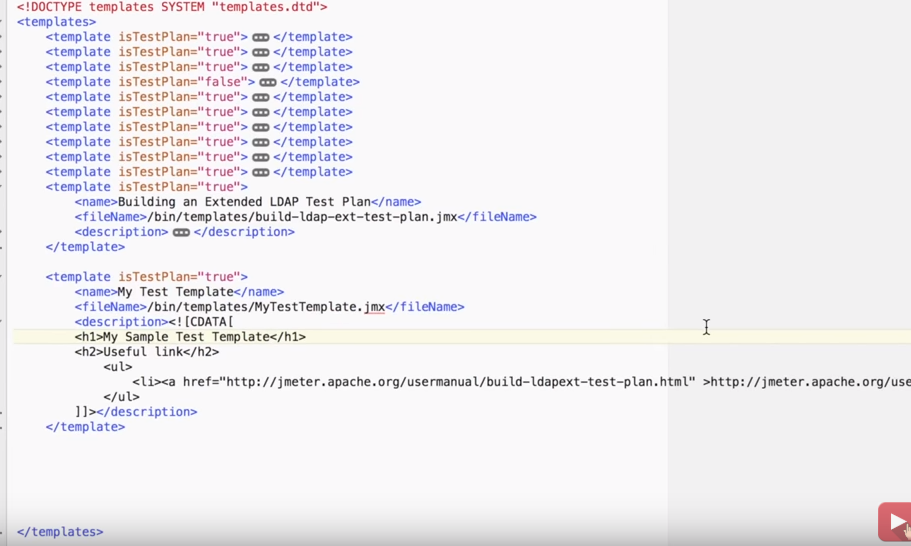
Select your template and click on Create.

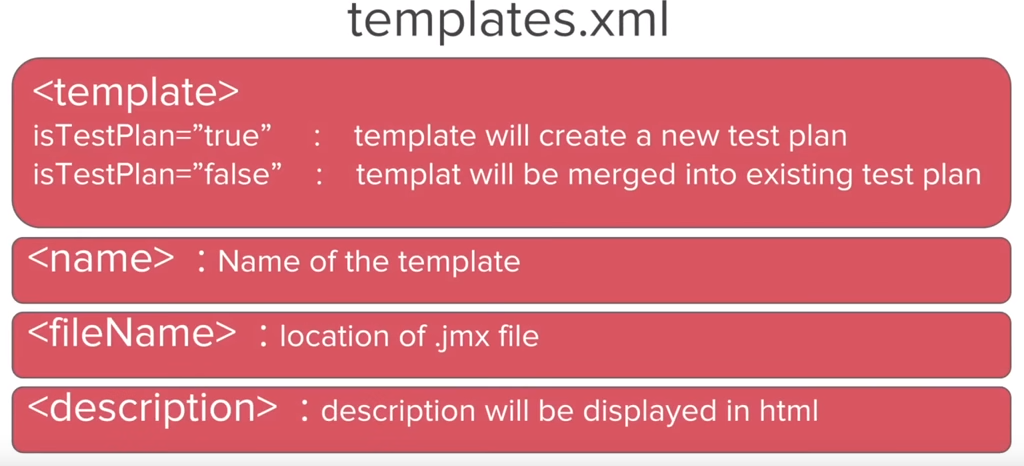
This will create a simple test plan with the necessary components.

To create you own template, create you test plan and Save As in the bin\template folder

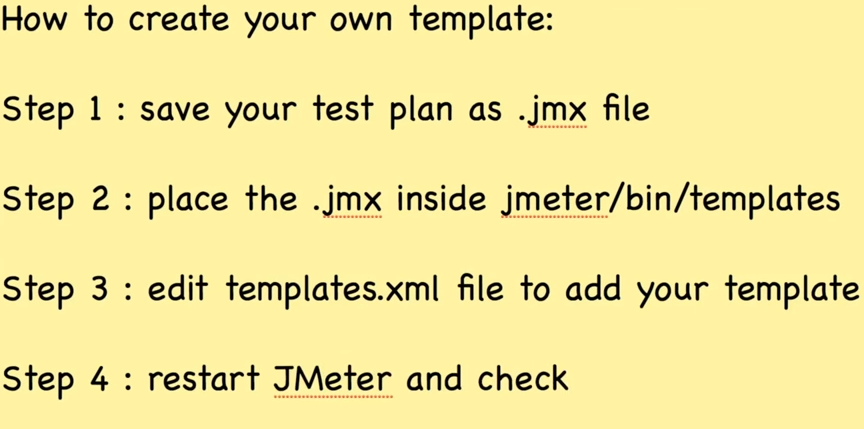


You then need to go into your templates.xml file in the bin\templates folder and edit it to contain the information of your new template. You need to at least update the name and the location.



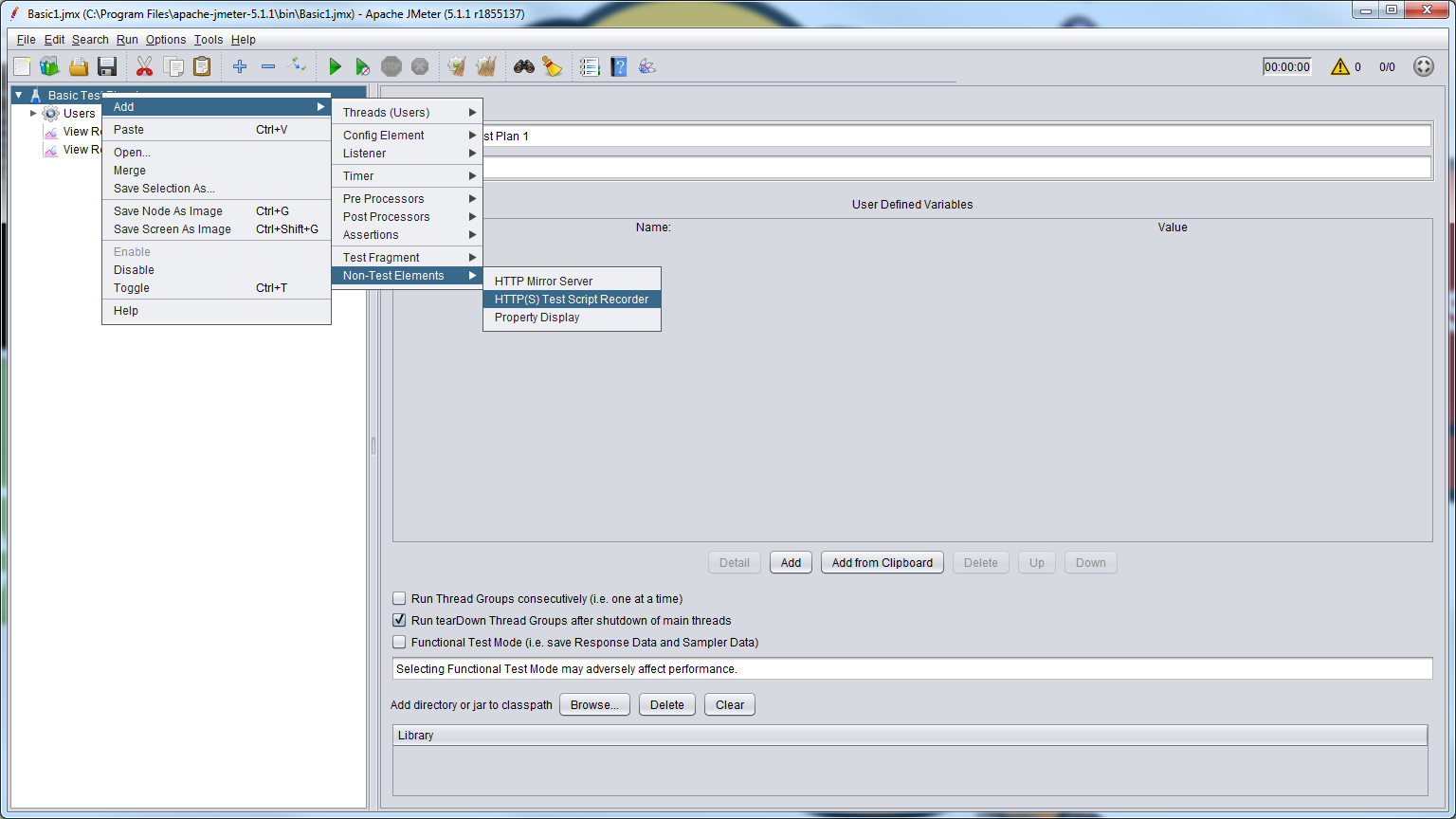


### In Summary

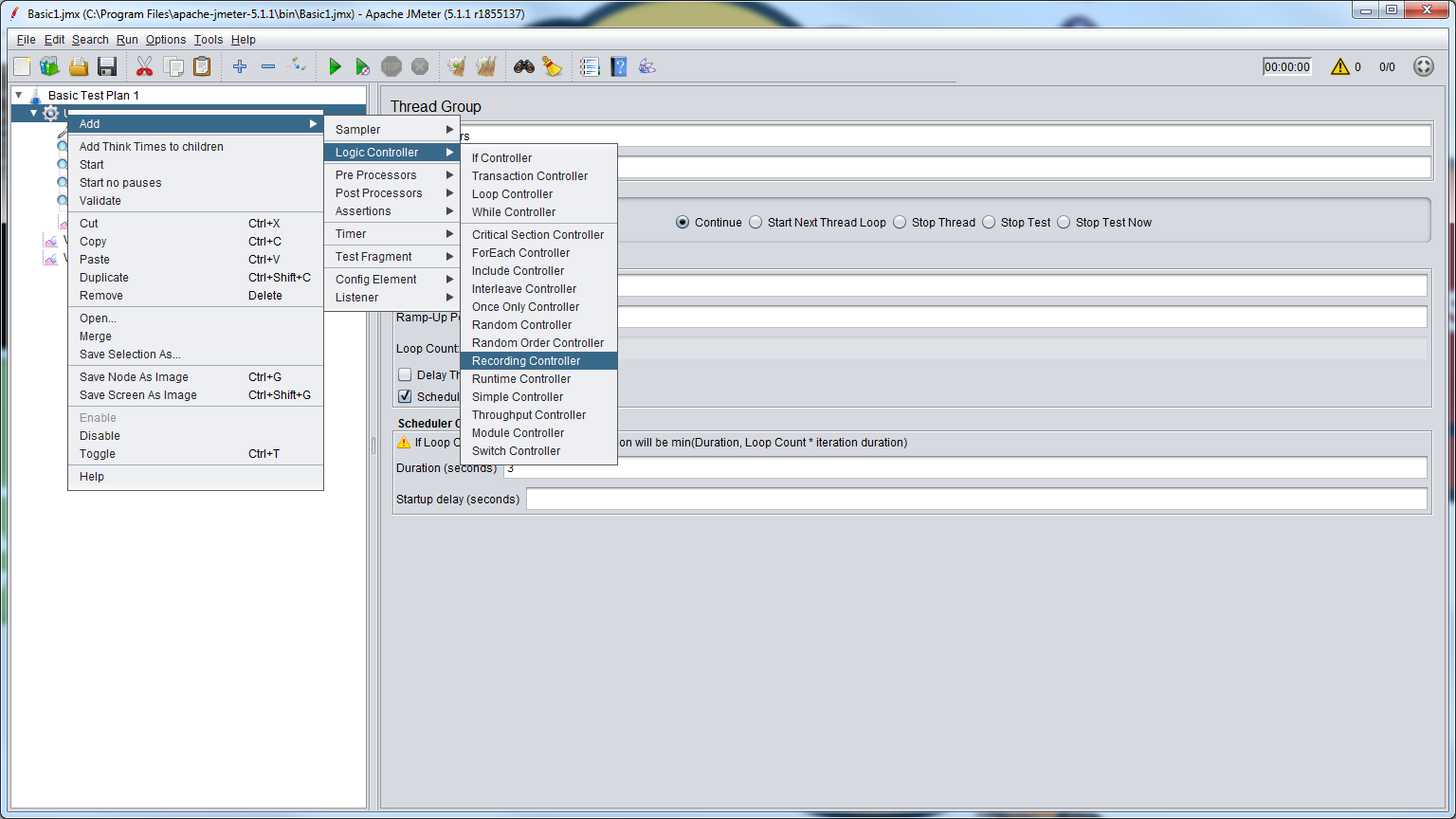


## Using Test Script Recorder

Right click on your test plan and select Add-> Non-Test Elements -> Test Script Recorder



Then go to your Thread Group and select Add-> Logic Controller – > Recording Controller



In the Test Script Recorder, you can just accept all the default values.

If want more information on these options then you can look in the JMeter online manual.

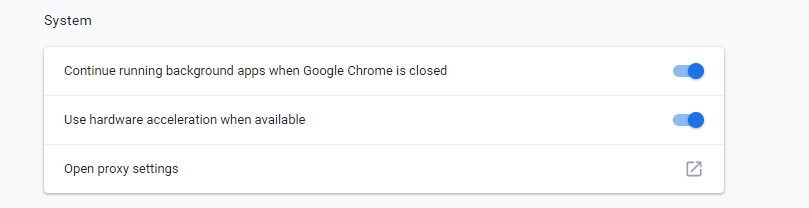
Before you start to record, you will need to go into your browser settings to set the port to be the same as we have in JMeter

For Chrome:

Click on 3 dots on top right of screen and select Settings.

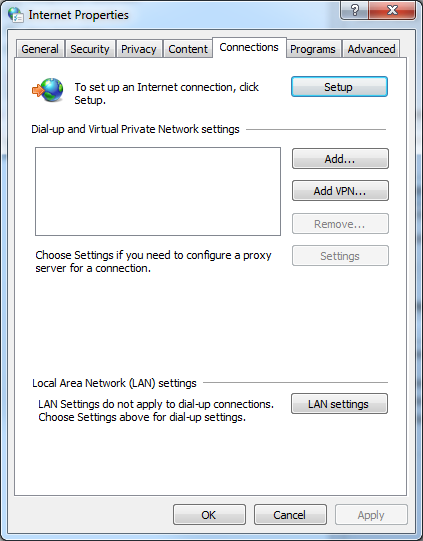
Scroll down to the bottom and click on Advanced.

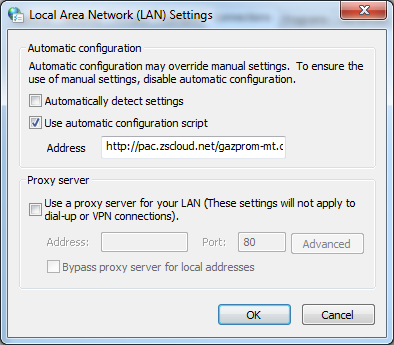
Scroll down again until you reach System.



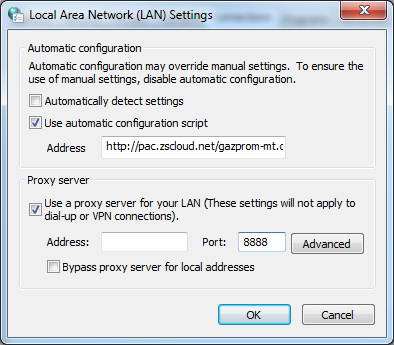
Click on Open proxy settings

Click on LAN settings





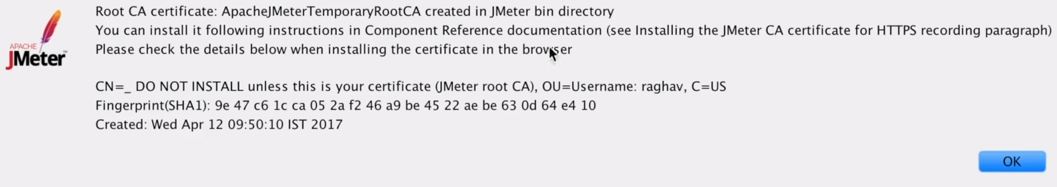
Set the proxy port to match that used in Jmeter Script Recorder screen.



Click on OK

Now run the recorder.

If you get a message like this:



You will have to add the generated certificate to your browser.

For Chrome:

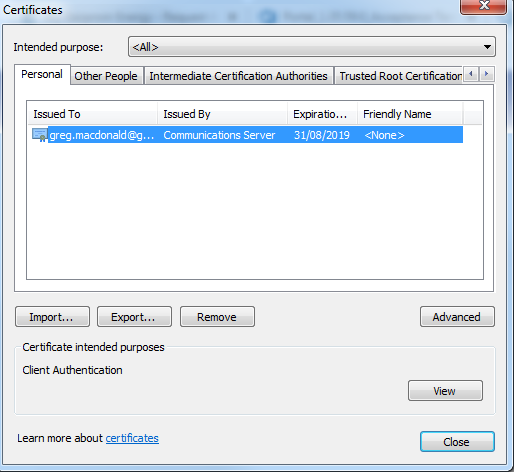
Click on 3 dots on top right of screen and select Settings.

Scroll down to the bottom and click on Advanced.

Scroll down again until your reach Manage certificates



Click on here them Import



Follow the steps on the certificate import wizard to import the certificate created in your Jmeter bin folder.

You can now start your recorder and it should save into the selected recording controller.

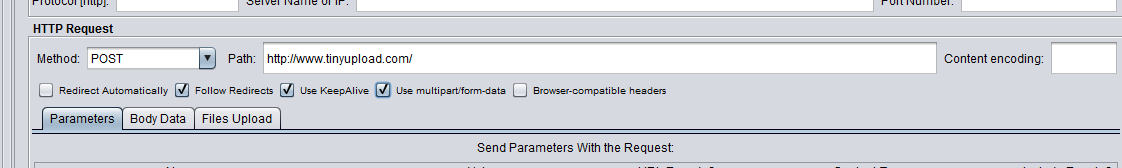
This does not seem to work, looks like manual proxy servers are being overridden.

If we need to use recording then will need some way of getting round this.

## How to Test File Upload

Create an HTTP Post request to the URL where you are uploading the file to.

You also ned to check the use multipart/form-data check box.



Click on the Files Upload tab and then the Add button

Click on Browse button and select the file you want to upload.

Inspect the web element of the upload button on the website and get its name.

(Make sure that the element you have selected has input type = “file”)

Past the name in the parameter Name field.

Next you will need the mime type of the file you are uploading

Go to <https://www.freeformatter.com/mime-types-list.html>

Find the MIME type for you file and past this into MIME Type field.

Add a listen to your HTTP request

Save your Test Plan and then run it.

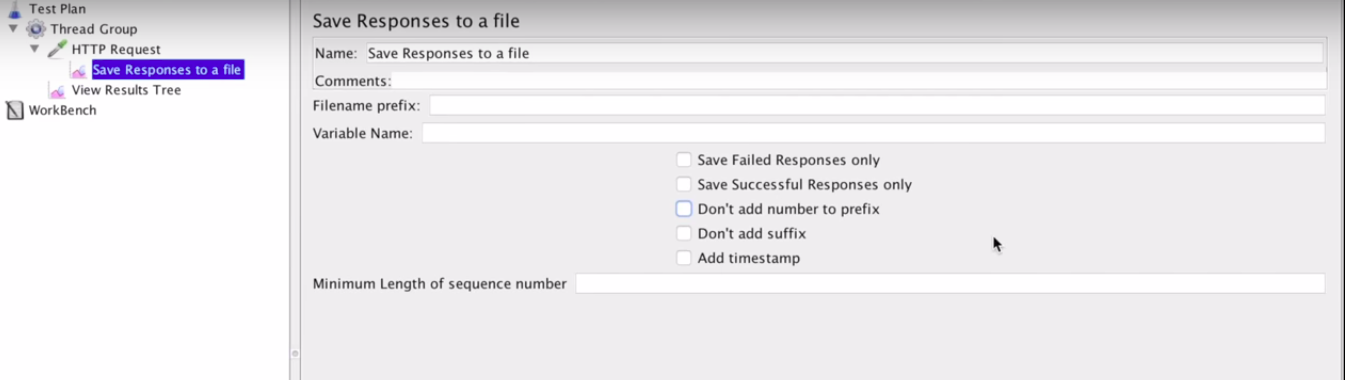
## How to Test File Download

Create Test Plan, Thread Group and HTTP Request (sampler) as normal

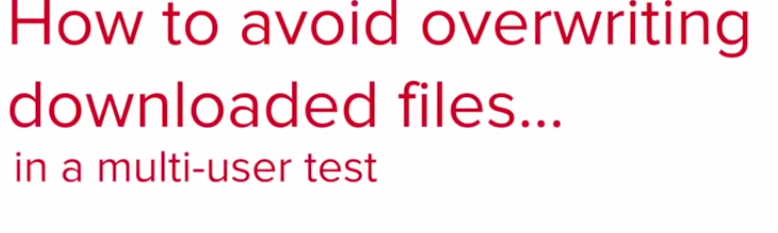
In the path in the HTTP Request, you need to add the path of the file location (not the download button, but location of the file). The request Method is a GET

Add a Listener (e.g. View Results Tree) to check that request was successful

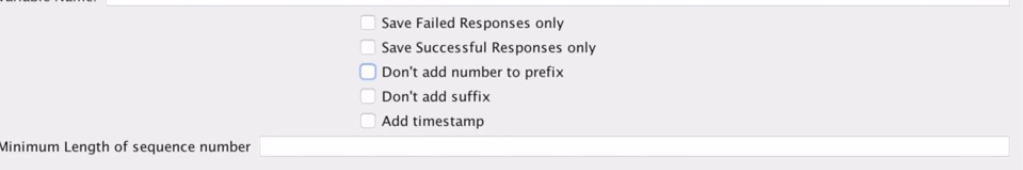
To check that the download of the file was successful, you need another Listener, ‘Save Response to a File’



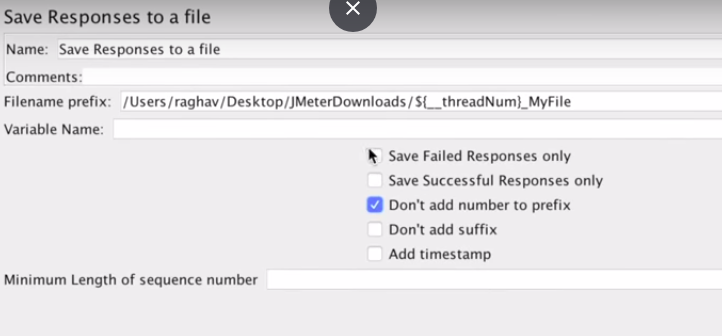
Filename prefix = location of where to save the downloaded file.



Uncheck the Don’t add number to prefix option



Alternatively, you can add the thread number as part of the filename instead of a prefix number



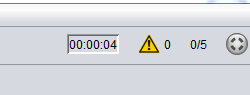
## Debugging

**View Results in Tree**

Your first port of call in debugging should be to add a View Results in Tree listener. This allows you to look at Sampler (Request) Result, Request Body and Headers and Response Body and Headers.

**Log Viewer**

You can access the Log Viewer from the yellow triangle at the top of the screen

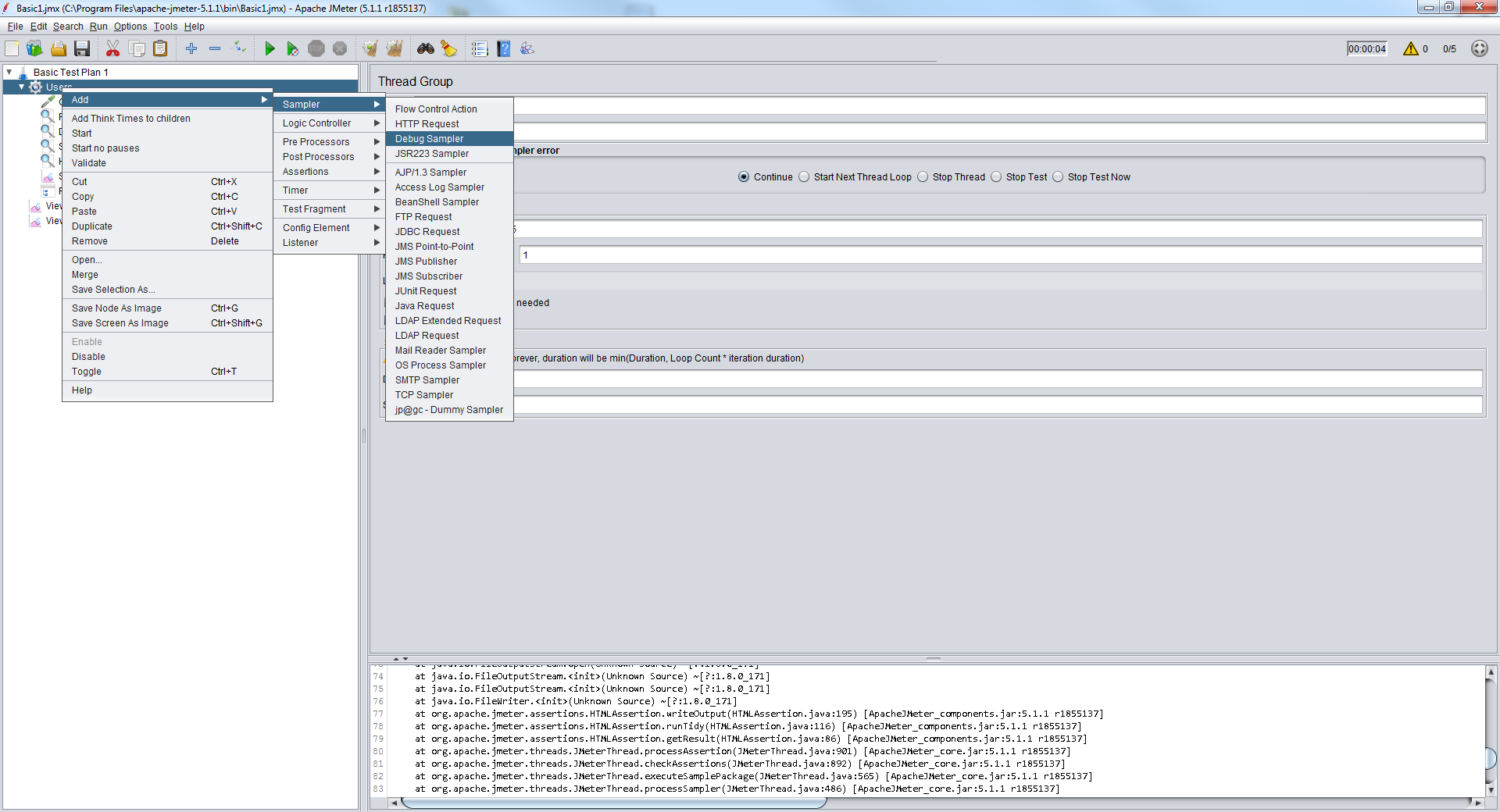


Or from Options->Log viewer in the main menu

**Debug Sampler**

You can add a Debug Sampler that gives information about you variables and properties at runtime.

Right click on your thread group and select Add -> Sampler -> Debug Sampler.

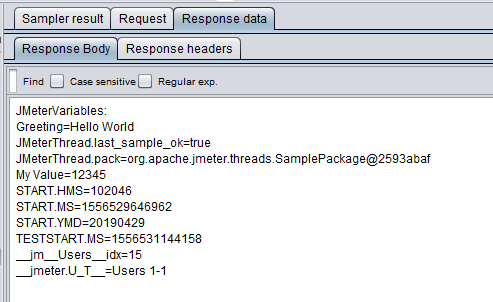


Make sure you have added the View Results Tree listener as the Debug Sampler results are only available in this listener.

To see this in action, add some variables to your Test Plan



Run your request and then you can see these variables in the Response data -> Response Body

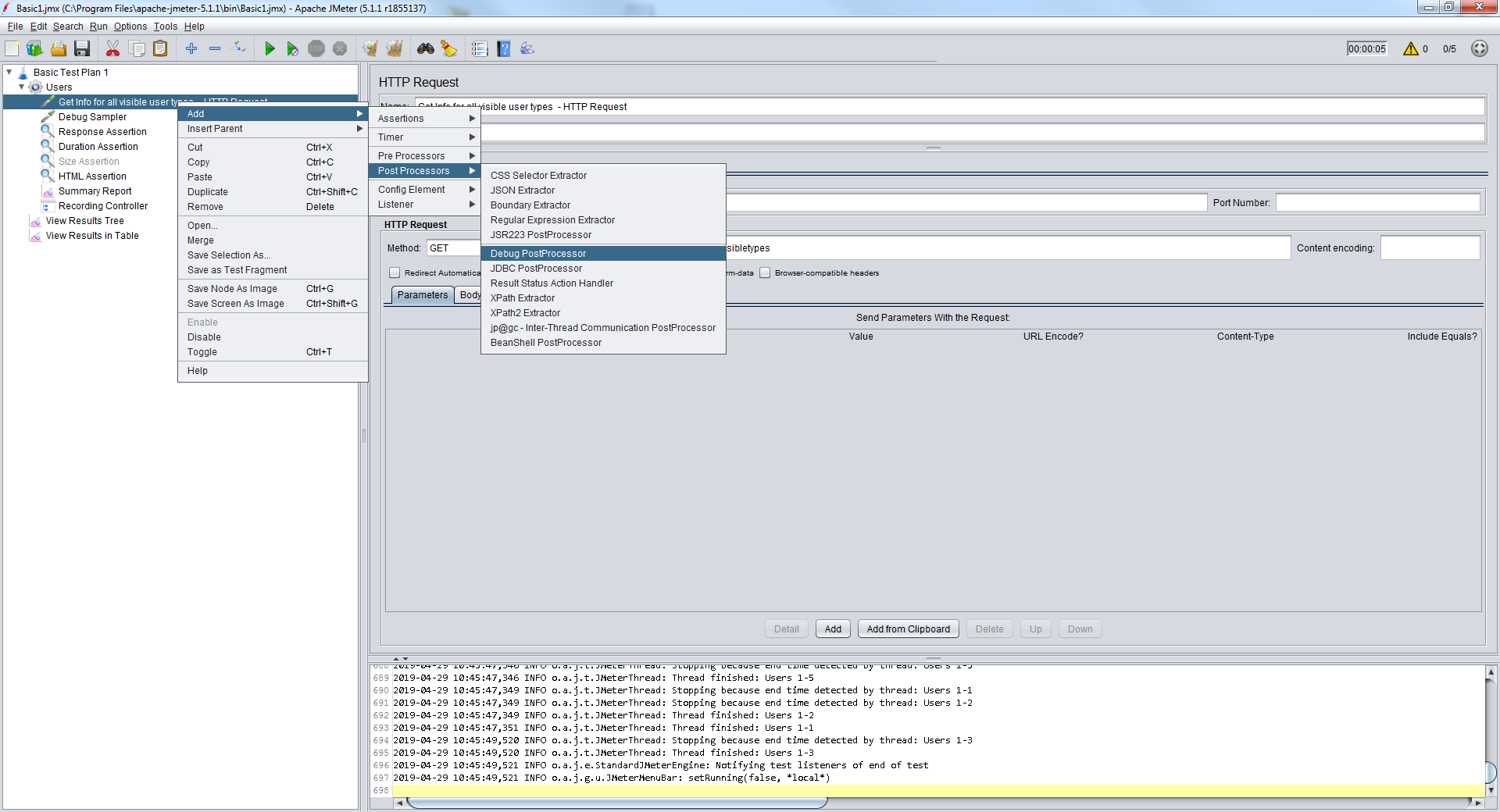


Remember to disable all these listeners and debuggers when you are running you tests outside of debugging, as they are all memory intensive.

**Debug PostProcessor**

The Debug Sampler works at the Thread Group level, so its scope is all your requests within the thread group. If you want to debug for a specific request, then you need a Debug PostProcesssor.

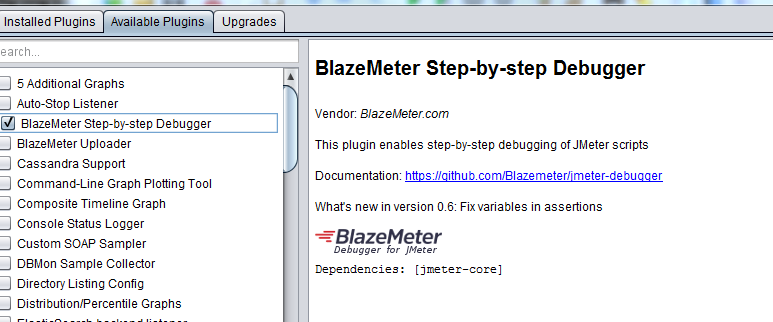
Right click on your request and select Add-> Post Processors -> Debug PostProcessor



You can also enable debug for a specific element. Doing this will cause some extra information for that element to be included in Debug logs. Select the element, then go to the main menu and select Help -> Enable Debug.

**BlazeMeter Step-by-Step Debugger.**

For even more detailed debugging (if you have lots of complicated scripts) you can enable the BlazeMeter Step-by-Step Debugger from the Plug in Manager screen.



Once installed, this can be accessed via the Run menu. There will also be an icon on the icon ribbon for it.

## How to record a test with Blaze Meter

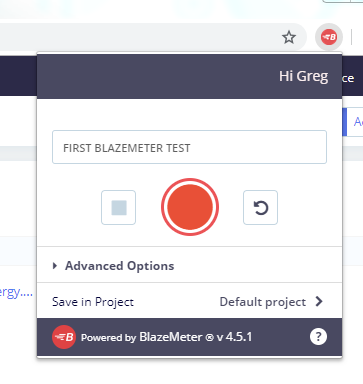
Download Blaze Meter extension from Chrome Store. Once done you will see this icon at the top of the screen.



Sign up to create Blaze Meter account and then sign in .

**Record a Test**

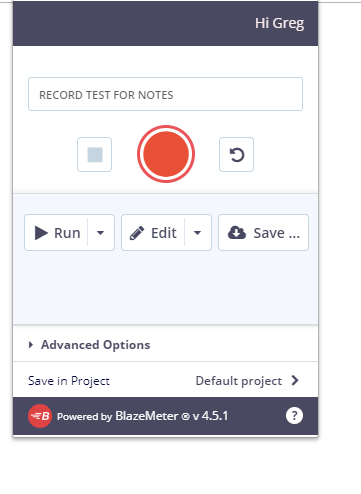
Click on the Blaze Meter icon at top of the screen.

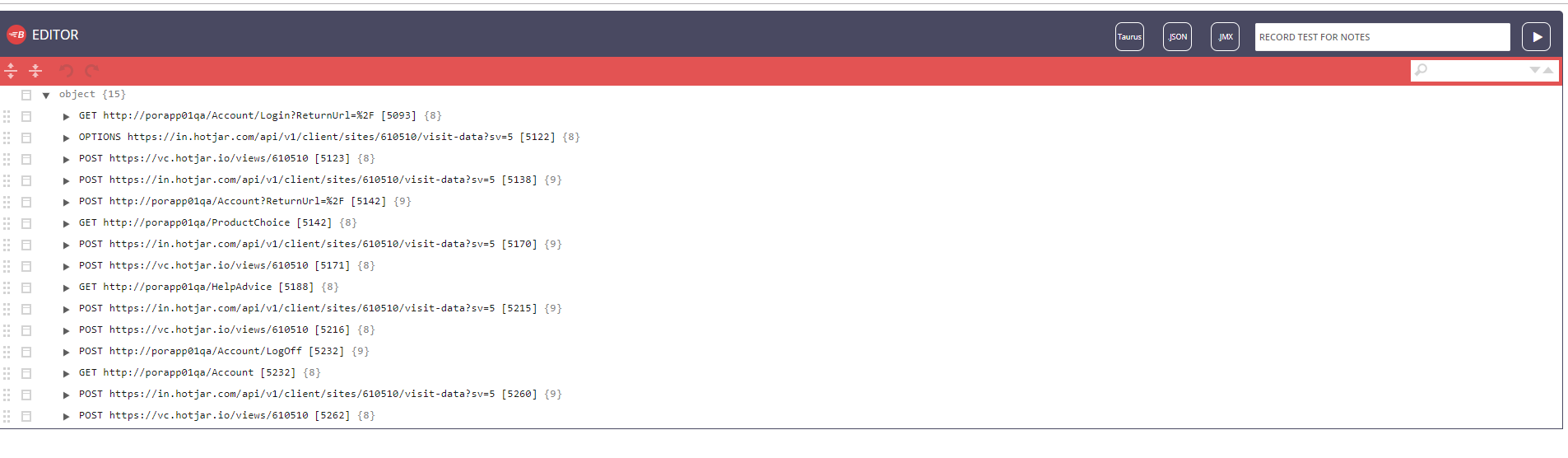


Give it a name and click on the round orange icon.

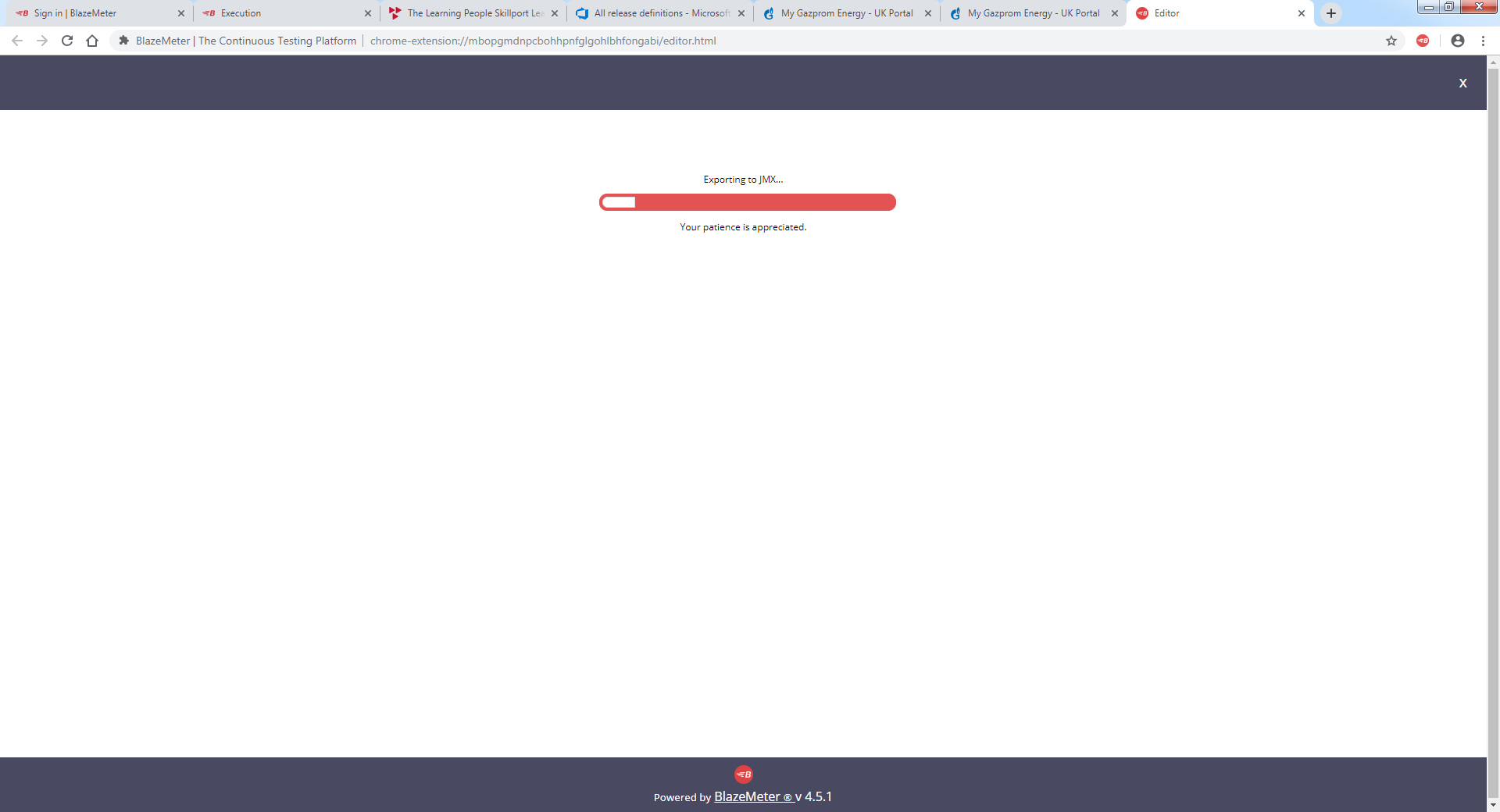
Perform the steps that you want to record then click on Save button.

Once done, you an click on Edit and select JMeter Script





Click on JMX to export to Jmeter.



Copy the downloaded file to the required location in your Jmeter folder.