Vancouver Software Test Automation Group

- Our goal: To ensure quality software by incorporating automated processes to help drive continuous integration.
- Call for presenters – your chance to show the group something cool you're working on!

Load/Performance testing with JMeter

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes
1	19:55:49.568	Thread Group 1-1	Search on Google	1868		60987
2	19:55:51.491	Thread Group 1-1	Search on Google	1022		63320
3	19:55:52.521	Thread Group 1-1	Search on Google	1090		62216
4	19:55:53.616	Thread Group 1-1	Search on Google	1184		48843
5	19:55:54.805	Thread Group 1-1	Search on Google	684		64026
6	19:55:55.495	Thread Group 1-1	Search on Google	1197		60336
7	19:55:56.698	Thread Group 1-1	Search on Google	683		61774
8	19:55:57.387	Thread Group 1-1	Search on Google	516		63171
9	19:55:57.909	Thread Group 1-1	Search on Google	470		48756
10	19:55:58.383	Thread Group 1-1	Search on Google	576		63626
11	19:55:58.965	Thread Group 1-1	Search on Google	468		60331
12	19:55:59.437	Thread Group 1-1	Search on Google	534		61866
13	19:55:59.978	Thread Group 1-1	Search on Google	528		63091
14	19:56:00.512	Thread Group 1-1	Search on Google	468		48754
15	19:56:00.985	Thread Group 1-1	Search on Google	1247		63137
16	19:56:02.239	Thread Group 1-1	Search on Google	462		60334
17	19:56:02.707	Thread Group 1-1	Search on Google	455		61864
18	19:56:03.167	Thread Group 1-1	Search on Google	464		63167
19	19:56:03.637	Thread Group 1-1	Search on Google	393		48832
20	19:56:04.035	Thread Group 1-1	Search on Google	495		63134
21	19:56:04.536	Thread Group 1-1	Search on Google	425		60336
22	19:56:04.967	Thread Group 1-1	Search on Google	464		61857
23	19:56:05.436	Thread Group 1-1	Search on Google	458		63085
24	19:56:05.900	Thread Group 1-1	Search on Google	469		48839
25	19:56:06.374	Thread Group 1-1	Search on Google	537		63542
26	19:56:06.917	Thread Group 1-1	Search on Google	565		60248
27	19:56:07.527	Thread Group 1-1	Search on Google	408		61857
28	19:56:07.940	Thread Group 1-1	Search on Google	501		63166
29	19:56:08.487	Thread Group 1-1	Search on Google	464		48839
30	19:56:08.956	Thread Group 1-1	Search on Google	488		63472

Sponsor shout outs

- Make / Dell Canada for the office space and refreshments.
- BlazeMeter, the JMeter cloud for providing free credits to run test scripts using their service.

Agenda

- Why is load/performance testing important?
- Introduction to JMeter.
- Working with JMeter configuration elements.
- Using the built-in HTTP Proxy server.
- Incorporating into the build process.
- Running tests from different regions of the world with BlazeMeter.

A little about me

- Software Developer at Make Technologies / Dell Canada.
- Passionate about all aspects of software development, including automation and ensuring a quality product.
- Interested in building great software.

Why load test?

- Can identify any bottlenecks before they are released to the public.
- Can also identify infrastructure issues.
- Can be used to tune caching and garbage collection parameters.
- Can establish baseline for maximum/sustained user load.

How load testing helped us at Make/Dell

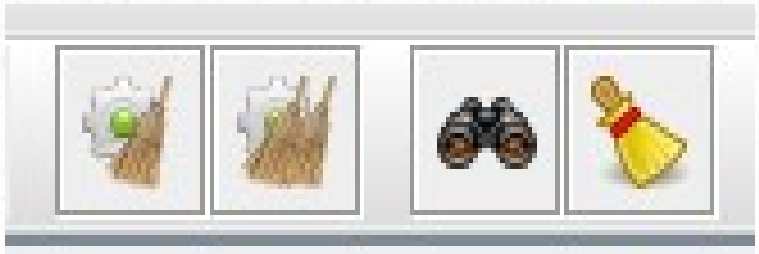
Helped us with identifying various sources from which bottlenecks originated.

We have a complicated set-up: load balancers, dynamic instances, MongoDB server, Neo4j server, Spring/Java web container.

Helped out with identifying vendor specific problems (non-thread safe access).

What is JMeter?

- JMeter is a GUI-based tool for creating and performing load tests.
- Can test multiple protocols – HTTP, FTP, JMS, etc.
- It's been around for awhile now – over 10 years since the first release.
- Check out the icons on the toolbar...



Why use JMeter?

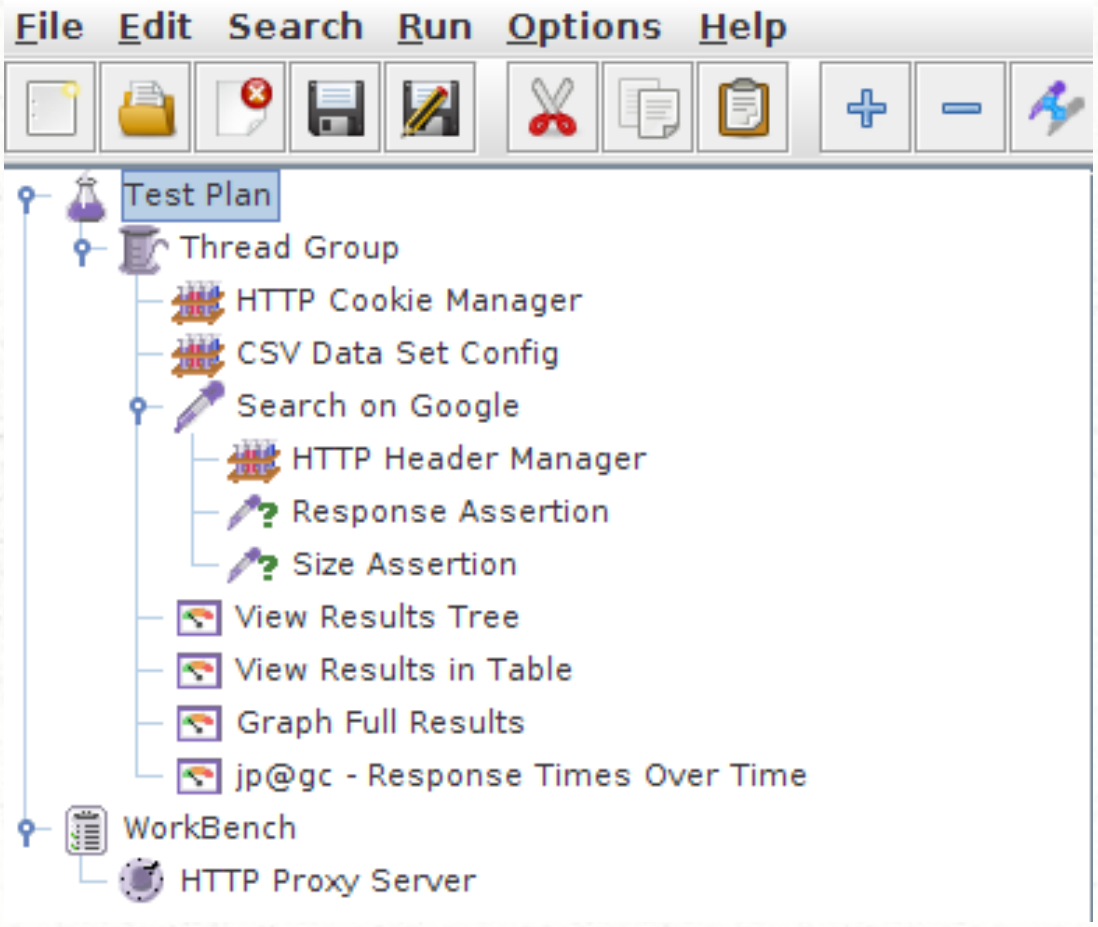
- It's free!
- It's (relatively) easy to use.
- Can be incorporated into the automated build process to monitor performance over time.
- 3rd party plug-ins for more fine grain control and reporting.
- Can be run on the cloud from different regions of the world.

Installation details

- It's as simple as downloading the tool and running it.
- Need the Java Runtime Environment

  http://jmeter.apache.org/usermanual/get-started.html

Anatomy of a test

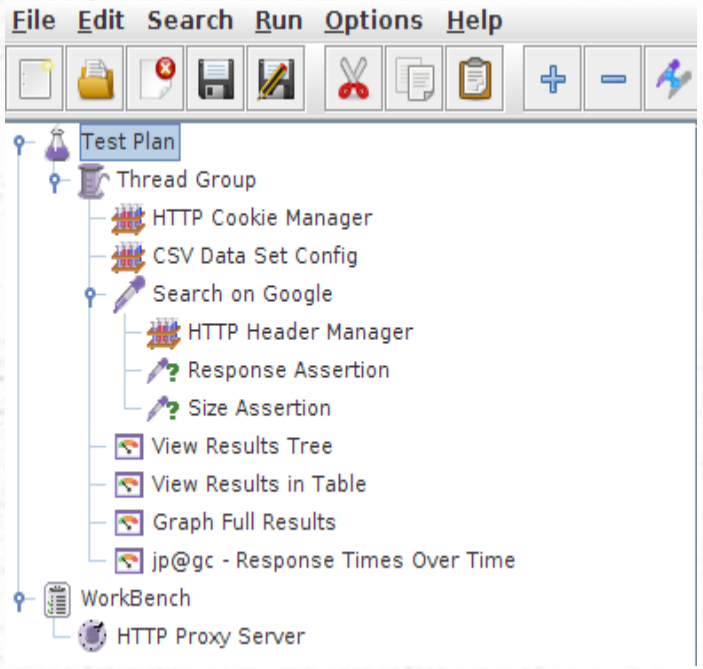


General structure of a test

- Start with a test plan.
- Add test cases to the test plan.
- Add elements to the test cases, which perform a variety of things.
- A given test could contain:
  - Configuration elements
  - Sampler elements
  - Assertion elements
  - Result listener elements

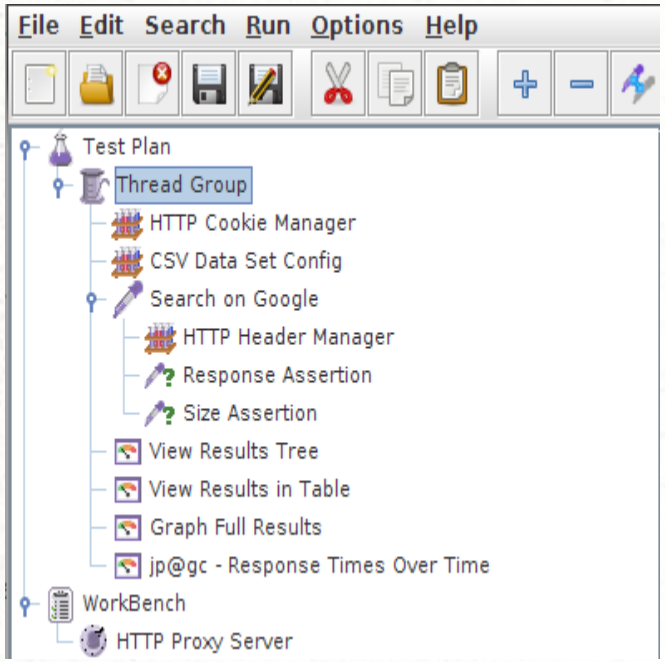
1. Test Plan

- Contains the tests which would be executed for each Thread Group.
- Can contain main configuration values to all tests.
- Think of a test plan as a container for test cases.



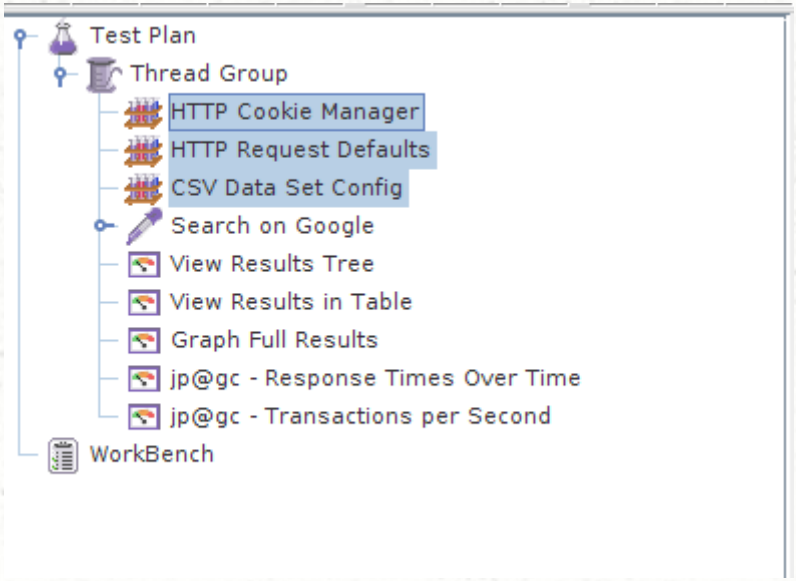
2. Thread Groups

- Thread groups are containers for individual steps of a test.
- They declare how fast the test case should run and how many requests the test case should have.
- Think of a thread group as a test case.



3. Configuration Elements

- Configuration elements provide additional support for the test cases.

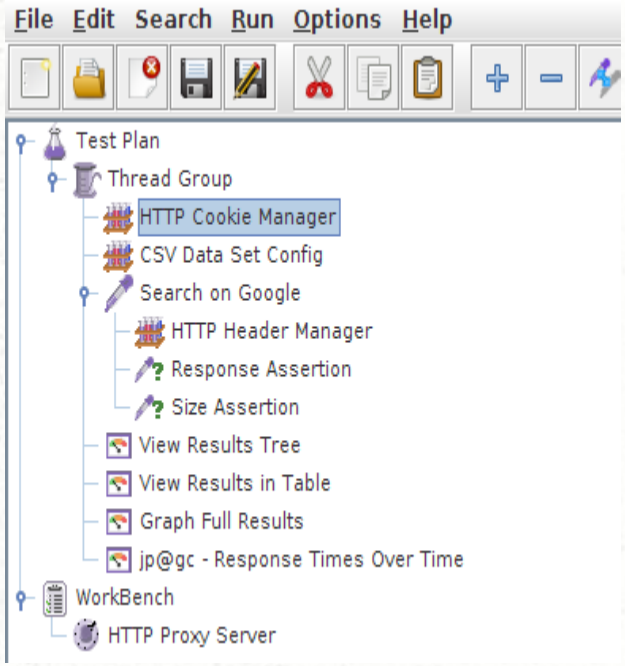


3. Configuration Elements

- Most common ones:
  - HTTP Cookie manager – for storage of cookies generated by server for the duration of the test case.
  - CSV Data Set Config – for reading from a csv sets of predefined values.
  - HTTP Request Defaults – set up the default base url, protocol and port.

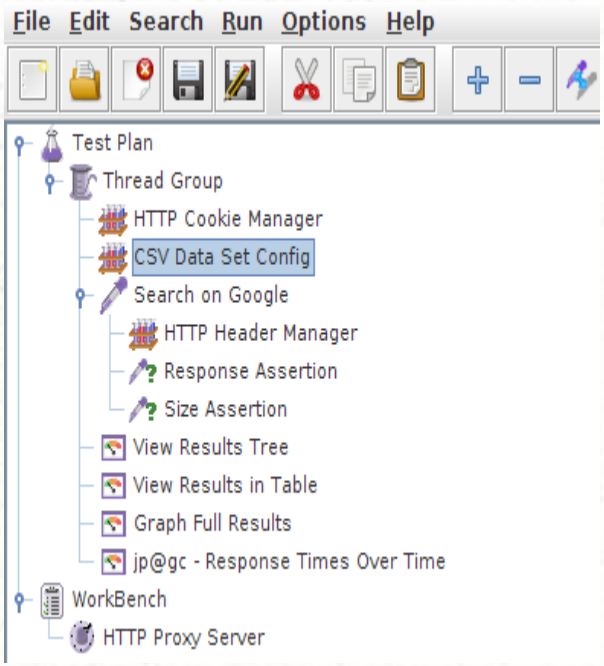
HTTP Cookie Manager

- Stores cookies that were created from the HTTP requests.
- User can also add predefined cookies.
- You have to add this configuration element in order to “enable” cookies!



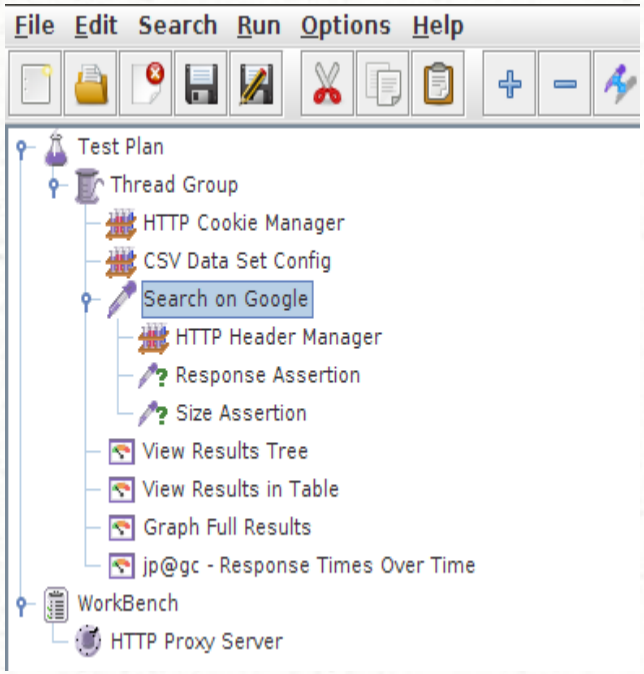
CSV Data Set Config

- This configuration element lets you read values from a CSV file and use it in the test case.
- Provides options to stop test after all values are read and used.
- CSV file read is relative to the JMeter test file location.



4. Request Samplers

- Samplers are the actions that perform a task.
- In this case it makes an HTTP request to the server.
- JMeter includes a lot of other samplers, but we only use the HTTP related ones.

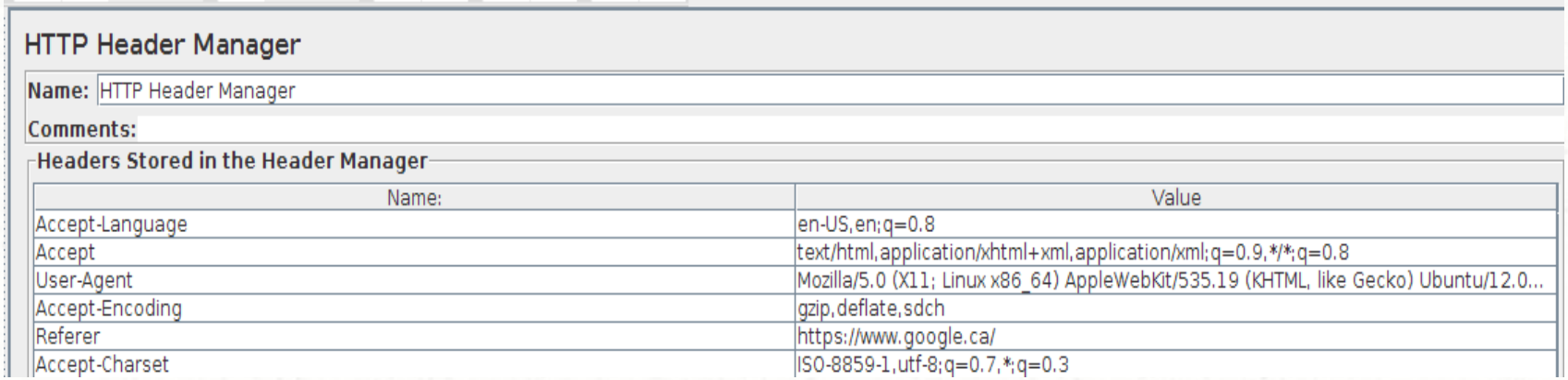


HTTP Sampler

- Most of the requests are done using HTTP Samplers.
- Can add parameters to the request.
- Can use variables with ${varName}.

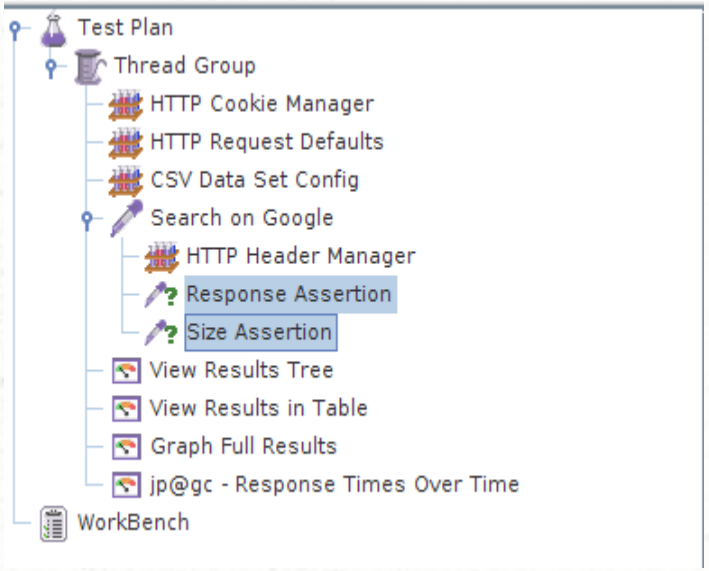
HTTP Sampler (cont.)

- Can optionally add a header to the HTTP request to mimic a real browser request.
- Required for some servers.



HTTP Sampler (cont.)

- Can also add an assertion after sampler is taken.
- Most common assertions:
  - Size Assertion
  - Response Assertion

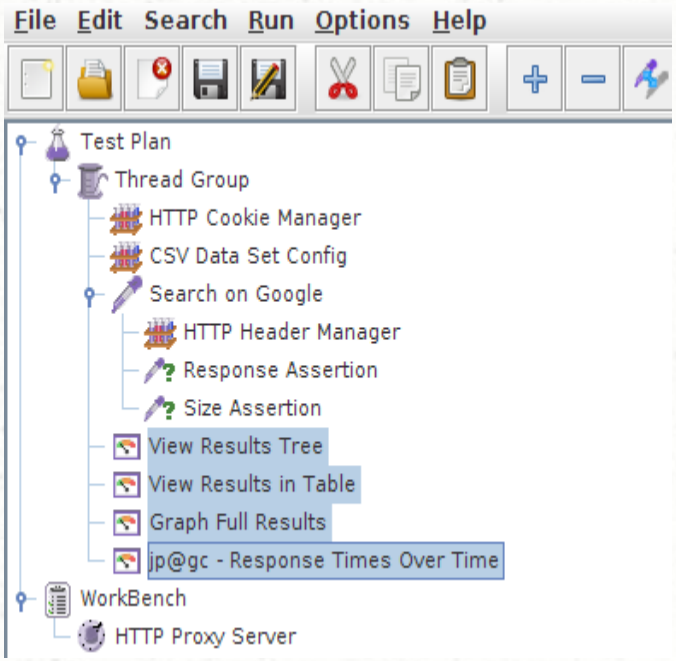


Assertions

- Size Assertion – Used to determine that the response is over/under/equal a certain size.
- Response Assertion – Used to match what we expect to the actual response.
- Expected values can be parameterized (read from CSV).

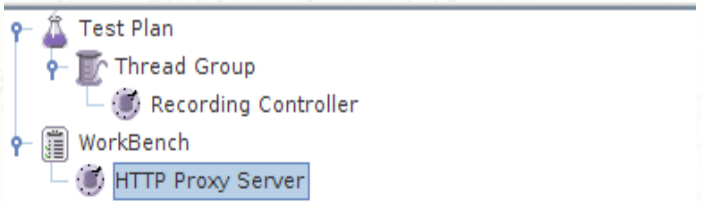
Result Listeners

- Result listeners are used to monitor the results as the test results come in from the samplers.
- Can view individual responses as well as avg/min/max times.
- Can also utilize plugins for extra reports.



HTTP Proxy Server

- JMeter has a nifty tool for "recording" HTTP requests via the HTTP Proxy Server.
- Add it under the "Work Bench" section of the tool.
- Add a Recording Controller in your test case to collect the requests.



HTTP Proxy Server

- Good for establishing a set of base line requests.
- It's a good idea to rename the requests for readability later on.
- It's also a good idea to add assertions to requests.
- As always, capture what varies into a CSV file (It's easier to change a CSV file than a test).

Install the jp@gc plugin

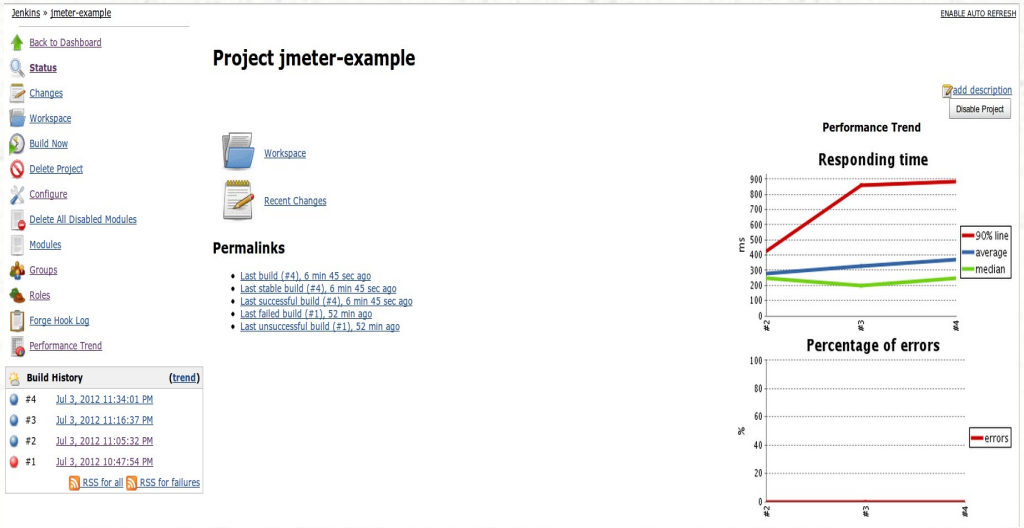
- Awesome plug-in for JMeter.
- Provides fine grained control over ramp up/down profiles, extra graphs/metrics and more!
- Read more from the latest edition of Methods and Tools magazine: http://www.methodsandtools.com/
- Download and install the plugin: http://code.google.com/p/jmeter-plugins/

BlazeMeter plugin

- Optional plug-in that integrates with BlazeMeter.
- Extra analytics including comparing performance over previous test runs.
- Check it out at http://www.blazemeter.com

Integrating the build process

- Can use Maven to incorporating into a Hudson/Jenkins server.
- Jenkins/Hudson Performance plug-in can parse JMeter result files.



Tips and advice

- Try to have assertions for each HTTP request – provides sanity check.
- Create tests only when development phase is almost complete.
- Try not to hard code values. Read them from a CSV file instead.

Advanced JMeter topics

- Extract variable from a request and use it in a subsequent request.
  - Use a Random variable in your requests.
  - Transaction Controllers.
  - Using the result of a previous request.
  - Incorporating JMeter tests into a Jenkins server (using Maven).
- Above examples usage and explanations are on my Github account, under tommytcchan/jmeter-presentation

Running tests on the cloud

- BlazeMeter, an out-of-the-box performance and load testing cloud, 100% compatible with JMeter.
- Can upload your tests to their server, and they will run your tests from supported regions in the world: US East, US West, Brazil, Japan, and more!

Demo

- Let's see a demo result from a run on BlazeMeter..

Q + A

- Questions?
- Comments?