

Seção 01: JMeter Tutorial Inicial - 01 Instalação

- 1- Verificar se o Java está instalado no sistema (cmd: java -version) → Java 7 =>
- 2- Download JMeter
- 3- Deszip JMeter
- 4- Start JMeter – jmeter/bin/jmeter.bat

- 02 Criar Primeiro Teste Jmeter

- 1- Jmeter.bat
- 2- Test Plan → Container que contém todos os elementos
- 3- Botão direito/ add/ Threads(Users)/ Thread Group

→ Thread Group é usado para criar ou iniciar os passos

Opções: Thread Properties – parte mais importante

- Number of Threads: Users simulados para o teste
- Ramp-Up Period(s): Tempo de execução
- Loop Count: opção de forever ou de quantidade
- Delay Thread creation until needed: atraso de passos
- Scheduler: agendador

- 4- Botão Direito Thread/ add/ Sampler/ HTTP Request

→ Teste um website

Web Server

Server Name or IP: link.com (sem http)

Port Number: nº da porta se for necessário

Path: acesso a uma sessão do site (ex: /calendário/2019-outubro)

- 5- Botão Direito Thread/ add/ Listener/ View Results in Table
- 6- Botão Direito Thread/ add/ Listener/ View Results Tree

- 03 Declarações

→ **Assertions**

→ Check das respostas, Erros e Warning

- 1- Botão Direito/ add/ Assertions/ Response Assertion

Response Field to Test → Response Code

Vai checker as respostas gerais(Patterns to Test)

Tipos de Assertions enviados ao Response Assertion:

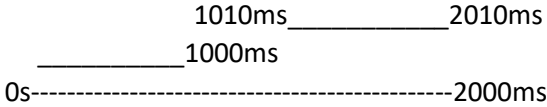
- 2- Botão Direito/ add/ Assertions/ Duration Assertion → Duração
- 3- Botão Direito/ add/ Assertions/ Size Assertion → Tamanho
- 4- Botão Direito/ add/ Assertions/ HTML Assertion → HTML
- 5- Botão Direito/ add/ Assertions/ XML Assertion → XML
- 6- Botão Direito/ add/ Assertions/ XPath Assertion → Teste Particular

- 04 Auditor/ Receptor/ Ouvintes

→ Listeners

→ Elementos que pegam/escutam informações dos testes que são executados.

→ Simples forma de visualizar os resultados e métodos do teste. Esses resultados mostram a performance dos testes.

- 1- Add: Thread Group e HTTP Request
- 2- Botão direito Thread Group/ Add/ Listener/ View Results in Table
→ Uma linha para cada Request
- 3- Mostra quantidade de acessos/usuários
→ Latência = tempo do primeiro byte/dado (1010ms)

0s-----2000ms
- 4- Botão direito Thread Group/ Add/ Listener/ View Results Tree
→ Importante notar que a lista em árvore(tree) ocupa muito espaço na memória. Ideal apenas para verificar a qualidade do seu teste e eficiência, depois o mais aconselhável é retira-lo.
- 5- Botão direito Thread Group/ Add/ Listener/ Aggregate Report
→ Única linha de resumo das respostas geradas
→ Median: tempo médio de espera das Requests
- 6- Botão direito Thread Group/ Add/ Listener/ Graph Results
→ Gráfico de performance e qualidade dos resultados
- 7- Botão direito Thread Group/ Add/ Listener/ Summary Report
→ Alguns dos mesmos resultados do Aggregate Report
- 8- Botão direito Thread Group/ Add/ Listener/ Simple Data Writer
→ registrar os Logs, registrar meus resultados em arquivos

Seção 2: Next Steps UI, Database, Command Line, FTP, API, JDBC - 05 Gravar UI Teste

- 1- Botão direito Test Plan/ Add/ Non-Test Elements/ HTTP(S) Test Script Recorder
 - Record UI Test
 - Workbench = HTTP(S) Test Script Recorder
 - Global Settings: port, é a porta particular que sera gravada e usada pelo teste
- 2- Tools(ferramentas) disponíveis para armazenamento de UI testes
 - a. Tools badboy version 2.2.5
 - b. Blazemeter
- 3- Use uma das ferramentas para fazer o mapa do teste Jmeter
 1. Escolha um serviço de teste e copie a Url na barra superior do Badboy, e inicie.
 2. Navegue pela página no próprio Badboy ele testara todos os caminhos acessados e visitados.
 3. Salve em arquivo Jmeter(.jmx) Script, exportando
- 4- Abra o Script no Jmeter
- 5- Botão direito Thread Group/ Add/ Listener/ View Results in Table
- 6- Botão direito Thread Group/ Add/ Listener/ View Results Tree
- 7- Start e Validar

- 06 Criando Database e Plano de Teste

- 1- Adicionar MySQL jdbc a pasta lib do Jmeter (Restart Jmeter)
- 2- ... / add/ Threads(Users)/ Thread Group
- 3- Botão direito Thread Group/ add/ Config Element/ JDBC Connection Configuration
 - Mostra os detalhes do BD
 - Adicionar Database URL, JDBC Driver class, username e password.
- 4- Botão direito Thread Group/ add/ Sampler/ JDBC Request
 - Script e Requests SQLs separadas
 - SQL Query: Espaço de seleção de tipo e para comandos SQL, como Selects, Triggers e QuerySelectors
- 5- Botão direito Thread Group/ Add/ Listener/ View Results in Table
- 6- Botão direito Thread Group/ Add/ Listener/ View Results Tree
- 7- Start e Validar

- 07 Jmeter com Command Line (non-GUI mode)

→ Execute non-GUI mode:

- Consome mais memory/pesquisas
- Não recomendado para Teste de grande e carregamento
- Command Line pode ser integrada a outros sistemas (Jenkins, CI, etc.)

- 1- Botão direito/ add/ Threads(Users)/ Thread Group
- 2- Botão Direito Thread/ add/ Sampler/ HTTP Request
- 3- Botão direito Thread Group/ Add/ Listener/ View Results Tree
- 4- Cmd
- 5- `$jmeter .sh` → write a log file
- 6- `$jmeter -n -t /local/do/jmeter/test/script -l /local/de/resultados/arquivos`
 - n → mode non-GUI
 - t → local Jmeter Script
 - l → local dos arquivos de resultados
- 7- `$jmeter -?` → Mostra todos as opções
→ (-h, --?, -v, -p, -q, etc.)

- 08 Test FTP Upload e Download

→ Use Binary mode? : usado para FTP de zip file

→ Save File in Response? : validação de teste e get the file quando necessita de uma performance apropriada de testes
Response surgira nas Views Results.

- 1- Botão direito/ add/ Threads(Users)/ Thread Group
- 2- Botão Direito Thread/ add/ Sampler/ FTP Request →GET
 - www.swfwmd.state.fl.us/data/ftp
 - Server Name or IP: ftp.swfwmd.state.fl.us
 - get(RETR)
 - Username e Password (e-mail aconselhado)
 - FileZilla conectar: arquivar arquivos nessa localização ou local do sistema
- 3- Teste FTP GET e validate
- 4- Botão Direito Thread/ add/ Sampler/ FTP Request →PUT
 - www.swfwmd.state.fl.us/data/ftp
 - Server Name or IP: ftp.swfwmd.state.fl.us
 - put(STOR)
- 5- Botão direito Thread Group/ Add/ Listener/ View Results in Table
- 6- Botão direito Thread Group/ Add/ Listener/ View Results Tree

- 09 Teste de Web Services(API)

Aplicação ← API → Interface

WebService – cliente ← API → Server

Ex: Cozinha ← Garçom → Mesas

REST | SOAP → diferença é o formato do protocolo no recebimento das mensagens

a) REST API

- 1- Botão direito/ add/ Threads(Users)/ Thread Group
- 2- Botão Direito Thread/ add/ Sampler/ HTTP Request

OU

Botão Direito Thread/ add/ Sampler/ SOAP XML-RPC Request
→ Webserver

Server Name or IP: endereço.api.open

→ Send Parameters with the Request

Add = [Name] [Value] Include Equals? V

- 3- Botão direito Thread Group/ Add/ Listener/ View Results Tree

b) SOAP API

- 1- Botão Direito Thread/ add/ Sampler/ SOAP XML-RPC Request
- 2- Adiciona detalhes do SOAP API Request

→ Soap/ XML – RPC Data

- 3- Botão direito Thread Group/ Add/ Listener/ View Results Tree
- 4- Botão direito Thread Group/ Add/ Listener/ View Results Table

- 10 criar Assertions JDBC(Database)

** 06 Criando Database e Plano de Teste*

- 1- Botão direito/ add/ Threads(Users)/ Thread Group
- 2- Botão direito Thread Group/ add/ Sampler/ JDBC Request
- 3- Botão direito Thread Group/ Add/ Listener/ View Results Tree
- 4- Botão direito Thread Group/ Add/ Listener/ View Results Table

- 1- Botão direito JDBC Request/ add/ Assertions/ Response Assertions
- 2- Adiciona variáveis ao SQL Query do JDBC Request

[select * from Student1]

Parameter values: []

Parameter types: []

Variable names: [Col1,Col2,Col3,Col4]

→ Apply to: Jmeter Variable [Col4_2] do Response Assertions

- 3- Botão direito Thread Group/ add/ Listener/ Assertion Results

Seção 3: REPORTS | PLUGINS | CSV DATA DRIVEN TEST | FUNCTIONS | VARIABLES

- 11 create HTML dashboard Reports command line

- 1- Botão direito/ add/ Threads(Users)/ Thread Group
 - 2- Botão Direito Thread/ add/ Sampler/ HTTP Request → index.html
 - 3- Botão Direito Thread/ add/ Sampler/ HTTP Request → pag2.html
 - 4- Botão Direito Thread/ add/ Sampler/ HTTP Request → pag3.html
 - 5- Botão direito Thread/ add/ Assertions/ Response Assertions
 - 6- Salva e Fecha
- Executa command Line
- `$jmeter -n -t "local do script/arquivo jmeter" -l "local do seu arquivo de resultados" -e -o "caminho de reports HTML"`
 - `$jmeter -g "local e csv file" -o "output Folder"`
 - Analise o HTML(Dashboard) Reports
 - Duvidas de criação do dashboard, documentação Jmeter mostra passo a passo e resolução ideal dos problemas emergentes
 - link: <https://jmeter.apache.org/usermanual/generating-dashboard.html>

- 12 Plugin Manager

- Install new Plugins
 - Remove old Plugins
 - Upgrade existing Plugins
 - Information on Plugins
- 1- Downloads/ Install new plugins
Arquivos JAR: <https://jmeter-plugins.org/wiki/PluginsManager/>
 - Adicionar plugin a pasta **ext**
 - Options/ Plugins Manager
Link de plugins Jmeter → <https://jmeter-plugins.org/>
 - Localizar plugins: apache-jmeter/ lib/ext
 - Adicionar plugin a pasta **ext**
 - 2- Para plugin ser executado, deve-se reiniciar a aplicação Jmeter.

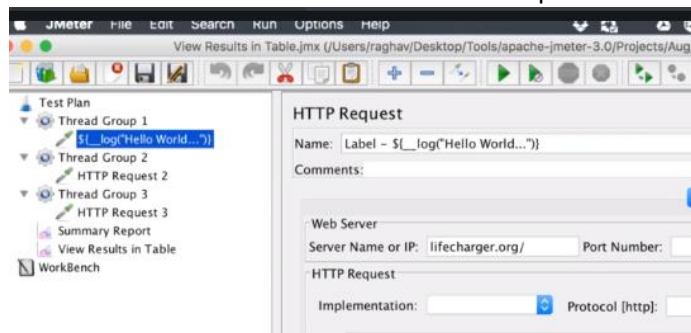
- 13 Leitura de Dados do Arquivo csv – Parametrização

* 09 Teste de Web Service(API)

- 1- Botão Direito no Thread Group/ Add/ Config Element/ CSV Data Set Config
 - a. Filename (rota/do/arquivos/csv.csv/excel)
 - b. Variable Names (comma-delimited)
 - c. Allow quoted data? (True/False/Edit{ })
 - d. Recycle on EOF = configuração de porta
- 2- Atualizar valor dos campos: `${ variable_name }`
→ URL value

- 14 Funções e Variáveis

- Criar 3 ThreadGroup com HTTP Request
- Criar Summary Report e View Results in Table
- Adicionar Function no name do HTTP Request



- **Function:** Método que preenche um campo e qualquer outro elemento no teste

`${ __functionName }`

`${ __functionName(var1, var2, ... , varN)`

- **Variable:** Aloca valores que podem referenciar qualquer elemento com o thread

Function – caseSensitive | came | Casing

- 1- Log: `label - ${__log("message")}`
- 2- Time: `label - ${__time(dd MM YYYY HH mm ss)}`
- 3- ThreadNum: `label - ${__threadNum}`
- 4- Soma: `label - ${__intSum(2,3,result)}`

Sample #	Start Time	Thread Name	Label	Sample Time...	Status	Bytes
1	21:14:42.946	Thread Group ...	HTTP Request 2	1287	✓	30594
2	21:14:42.947	Thread Group ...	HTTP Request 3	1951	✓	30587
3	21:14:42.943	Thread Group ...	5	2047	✓	30587
4	21:14:43.149	Thread Group ...	5	2159	✓	30594
5	21:14:43.140	Thread Group ...	5	2117	✓	30594
6	21:14:43.546	Thread Group ...	5	2363	✓	30594
7	21:14:43.747	Thread Group ...	5	2188	✓	30594
8	21:14:44.991	Thread Group ...	5	1634	✓	30594
9	21:14:45.309	Thread Group ...	5	1602	✓	30594
10	21:14:45.465	Thread Group ...	5	1616	✓	30594
11	21:14:45.936	Thread Group ...	5	1598	✓	30594
12	21:14:45.909	Thread Group ...	5	2644	✓	30594

Seção 4: ELEMENTOS DE USO FREQUENTE NO JMETER – 15 Setup Performances Reais

→ **Think Time** – Simula users atuais com timing/delay

→ **Pacing** – Controla os users virtuais

– Controla o tempo entre as interações

– Arquiva um número **n** de interações em **x** minutos/segundos

1- Adicionar Plugin – Stepping Thread Group

** 12 Plugin Manager*

2- Confirmar *jpgc – Standard Set*

3- BtnDireito TestPlan/ Add/ Threads(Users)/ jp@gc – Stepping Thread Group

4- ... / HTTP Request

5- ... / View Results Tree

6- ... / View Results in Table

7- ... / Aggregate Report

- 16 Timer, Como Adicionar Think Time

→ Pausar Thread (v.user) por algum tempo

→ Adiciona delay entre os Threads

→ Evitar sobrecarga no servidor e arquivar comportamento em tempo real passo a passo no carregamento

1- ... / Thread Group

2- ... / Add/ Logic Controller/ Simple Controller

3- BtnDireito Simple Controller/ ... / HTTP Request

4- BtnDireito Thread Group/ ... / View Results in Table

5- BtnDireito Simple Controller/ Add/ Timer/ Constant Timer

6- BtnDireito Simple Controller/ Add/ Timer/ Uniform Random Timer

7- Desabilitar todos os Constant Time

→ Random Delay Max

→ Constant Delay Offset

→ $0.X * \text{Random Delay Max} + \text{Constant Delay Offset}$

- 17 Parametrizar Teste FTP

* 8 Test FTP Upload e Download

→ Upload arquivos para o FTP com nomes diferentes

- 1- Adicionar CSV Data Set Config
 - Btn Direito Thread Group/ add/ Config Element/ CSV Data Set Config
- 2- Criar Arquivo CSV e prover localização no CSV Data Set Config
 - Adicionar aos Filename a rota do arquivo Excel
- 3- FTP Request PUT = Remote File: /rota/do/arquivo/ \${Name}
 - Parametriza FTP PUT

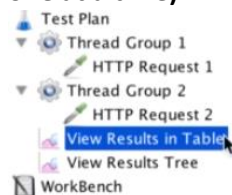
- 18 Iniciar execução Programada + Sequencial

→ Executar testes de Duração Específica

- 1- ... / Thread Group
 - Loop Counter **Forever** [v]
 - Selecionar a opção **Scheduler** [v]
 - Adicionar Duration (sec)
- 2- ... / HTTP Request
- 3- ... / View Results Tree
- 4- ... / View Results in Table

→ Executar testes Sequencialmente

- 1- Criar 2 Thread Groups (Thread Group1 e Thread Group2)
- 2- Criar HTTP Request (HTTP Request 1 e HTTP Request2) em cada TG (Thread Group)
- 3- Test Plan
 - Selecionar [v] **Run Thread Groups consecutively (i. e. run groups one at a time)**



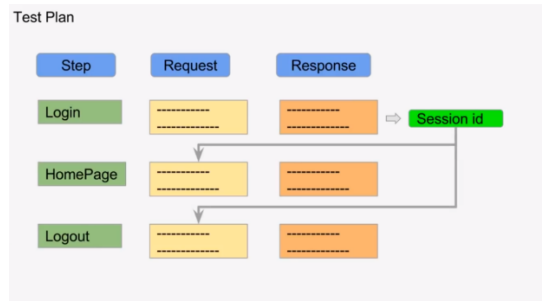
→ Adicionando Websites para testes Sequenciais

- 1- Criar 2 Thread Groups (Thread Group1 e Thread Group2)
- 2- Criar HTTP Request (HTTP Request 1 e HTTP Request2) em cada TG
- 3- Adicionar duração Máxima ao 1° TG (**30 segs.**)
- 4- Adicionar delay ao 2° TG (**10 segs.**)

- 19 Correlação (Expressão Regular com extração)

→ **Correlação:** Extração do valor da resposta de um Step e referência deste dentro de uma solicitação de outro Step subsequente.

Extrai dinamicamente o Valor da resposta do 1ºStep, por exemplo, e referencia esse valor extraído na solicitação do 2ºStep.



Referência: Feito dinamicamente – at runtime.

Step: Passo (1ºPasso, 2ºPasso, ...).

a) Uso de Expressões de Extração Regulares para Correlação

1- Criar um Test Plan

- ... / Thread Group
- ... / HTTP Request
- .../ View Results Tree (Pode-se usar o RegEx Tester)

2- Adicionar Regular Expression Extractor

- Onde o valor da resposta precisa ser extraído.
- BtnDir HTTP Request/ Add/ Post Processors/ Regular Expression Extractor

3- Referenciar o Valor Extraído ao Step subsequente

- regexr.com

- Adicionar o Nome da Referência: **ReferenceValue**

A imagem mostra a interface de configuração do 'Regular Expression Extractor'. O campo 'Field to check' tem 'Body' selecionado. O 'Reference Name' é definido como 'ReferenceValue'. A 'Regular Expression' é configurada como ''. O 'Template' é '\$1\$'. O 'Match No. (0 for Random)' é definido como '4'. O campo 'Default Value' está vazio.

- Adicionar outro HTTP Request com Nome da Referência Path:

`${ReferenceValue}`

A imagem mostra a interface para adicionar um novo 'HTTP Request' no 'WorkBench'. O caminho ('Path') é configurado como `${ReferenceValue}`, referenciando o valor extraído no step anterior. Outros campos como 'Web Server', 'Server Name or IP', 'HTTP Request' e 'Implementation' estão presentes, mas não foram preenchidos.

- 20 Uso de Templates

→ **JMeter Templates:** Scripts reutilizáveis. Usuário pode gerar e selecionar uma Templates de Test Plan com necessidades e componentes básicos.

Acesso as Templates

-> Barra de Ferramentas | File > Templates..

-> **Templates | Select Template:** [Recording, Recording with Think Time, JDBC Load Test, BeanShell Samples, MongoDB Load Test, Building a Web Test Plan, Building an advanced Web Test Plan, Building a SOAP WebService Test Plan]

-> Pode se fundir Templates. Abre-se um e logo em seguida abre-se outro, surgira a opção de Merge.

→ Criar Template

- 1- Salvar seu Test Plan como arquivo .jmx
- 2- Colocar seu Test Plan .jmx na rota jmeter/bin/template
- 3- Abrir o arquivo templates.xml e editar a lista de templates

```
<templates>
  <template isTestPlan="true"> </template>
  <template isTestPlan="true"> </template>
  <template isTestPlan="true"> </template>
  <template isTestPlan="false"> </template>
  <template isTestPlan="true"> </template>
  <template isTestPlan="true"> </template>
  <template isTestPlan="true"> </template>
  <template isTestPlan="true"> </template>
  <template isTestPlan="true"> </template>
  <template isTestPlan="true">
    <name>Building an Extended LDAP Test Plan</name>
    <fileName>bin/templates/build-ldap-ext-test-plan.jmx</fileName>
    <description> </description>
  </template>
  <template isTestPlan="true">
    <name>My Test Template</name>
    <fileName>bin/templates/build-ldap-ext-test-plan.jmx</fileName>
    <description><![CDATA[
      <h1>Test plan from Building an Extended LDAP Test Plan section in use
      <h2>Useful link</h2>
      <ul>
        <li><a href="http://jmeter.apache.org/usermanual/build-ldape
      </li>
    </ul>
    ]]></description>
  </template>
```

- 4- Reiniciar Jmeter e verificar

Seção 5: DEBUG E MAIS - 21 Teste Script Recorder

→ Workbench = HTTP(S) Test Script Recorder

→ Usado para graver ações no navegador

1- BtnDir. TG/ Add/ Logic Controller/ Recording Controller

→ Definir Configuração Proxy do Navegador (Tools/Options)

→ Instalar certificado no Navegador(bin/apacheJMetercertificate.crt)

2- BtnDir. HTTP(S) Test Script Recorder/ Add/ Listeners/ View Results Tree

→ Construir uma Template Recording (Recording with Think Time)

- Test Plan
 - ➔ User Defined Variables
 - ➔ HTTP Request Default
 - ➔ HTTP Cookie Manager
- Thread Group
 - ➔ View Results Tree
- HTTP(S) Test Script Recorder
 - ➔ View Results Tree

- 22 Teste de Upload de Arquivo (POST)

- 1- Criar um Test Plan de HTTP Request
- 2- Adicionar o *Server Name or IP*: (URL ou IP)
- 3- Alterar o *Method* de *GET* para *POST*
- 4- Selecionar o [] **Use multipart/form-data for POST**
- 5- Selecciona *Files Upload*
- 6- Clicar no botão *Add*
- 7- Clicar em *Browse..* e seleccionar o arquivo que foi feito Upload
- 8- No local onde ocorreu o Upload do arquivo, será necessário Inspeccionar o elemento e identicar seu *name* (ex: *name="upload_file"*)
- 9- Copiar o valor do *name*, colar no *Parameter Name*
- 10- No *freematter.com* verificar o caminho do seu tipo de arquivo
(Ex: *application/pdf* áudio/*x-aac* image/*png*)
- 11- Adicionar o caminho ao *MIME Type*

→ Recording File Upload

- 1- Templates > Recording = Create a Test Plan
- 2- HTTP(S) Test Script Recorder – Definir uma porta no *Port*
- 3- Definir Navegador com a *Port*
- 4- Preferences: Tools > Options > Advanced > Network> Connection Settings
- 5- **Start** HTTP(S) Test Script Recorder
- 6- Filtrar os resultados que você precisa
- 7- No View Results Tree/ Response Data verificar a mensagem do Script

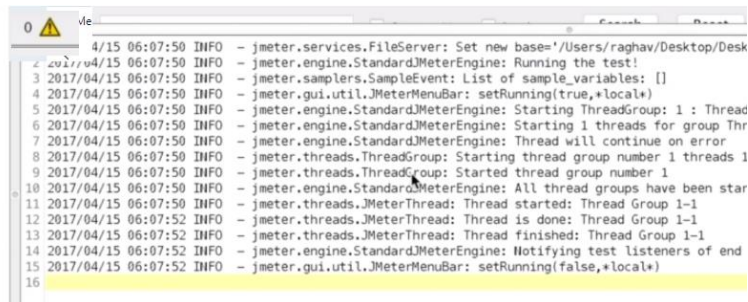
→ Verificar Problemas: Options/ Log Viewer

- 23 Teste de Download de Arquivo (GET)

- 1- Test Plan com HTTP Request
- 2- Alterar o *Method* para *GET*
- 3- Adicionar o *Server Name or IP*: (URL ou IP)
- 4- Adicionar Path (Ex: /1KB.zip)
- 5- Adicionar Listener/ View Results Tree
- 6- Btn dir. HTTP Request/ Add/ Listener/ Save Responses to a File
- 7- Save Responses to a File → Filename Prefix (caminho)
- 8- Selecionar **Don't add number to prefix [v]**
- 9- Options > Function Helper Dialog
 - a. Choose a Function `__threadNum`
 - b. Function String `${__threadNum}` como prefixo

- 24 DEBUG e Correção de erros

- 1- Adicionar um Test Plan com HTTP Request
- 2- Analisar View Results Tree Listener



```
0 4/15 06:07:50 INFO - jmeter.services.FileServer: Set new base='/Users/raghav/Desktop/Desk
1 2017/04/15 06:07:50 INFO - jmeter.engine.StandardJMeterEngine: Running the test!
2 2017/04/15 06:07:50 INFO - jmeter.samplers.SampleEvent: List of sample_variables: []
3 2017/04/15 06:07:50 INFO - jmeter.gui.util.JMeterMenuBar: setRunning(true,*local*)
4 2017/04/15 06:07:50 INFO - jmeter.engine.StandardJMeterEngine: Starting ThreadGroup: 1 : Thread
5 2017/04/15 06:07:50 INFO - jmeter.engine.StandardJMeterEngine: Starting 1 threads for group Thr
6 2017/04/15 06:07:50 INFO - jmeter.engine.StandardJMeterEngine: Thread will continue on error
7 2017/04/15 06:07:50 INFO - jmeter.threads.ThreadGroup: Starting thread group number 1 threads 1
8 2017/04/15 06:07:50 INFO - jmeter.threads.ThreadGroup: Started thread group number 1
9 2017/04/15 06:07:50 INFO - jmeter.engine.StandardJMeterEngine: All thread groups have been star
10 2017/04/15 06:07:50 INFO - jmeter.threads.JMeterThread: Thread started: Thread Group 1-1
11 2017/04/15 06:07:50 INFO - jmeter.threads.JMeterThread: Thread is done: Thread Group 1-1
12 2017/04/15 06:07:52 INFO - jmeter.threads.JMeterThread: Thread finished: Thread Group 1-1
13 2017/04/15 06:07:52 INFO - jmeter.engine.StandardJMeterEngine: Notifying test listeners of end
14 2017/04/15 06:07:52 INFO - jmeter.gui.util.JMeterMenuBar: setRunning(false,*local*)
15
16
```

- 3- Btn Dir. Thread Group/ Add Sampler/ Debug Sampler
 - Desabilitar o Debug Sampler antes de iniciar a performance teste
 - Btn dir. Debug Sampler /Disable
- 4- Btn Dir. HTTP Request/ Add/ Post Processors/ Debug PostProcessor
- 5- Btn Dir. HTTP Request/ Add/ Config Element/ HTTP Cache Manager
- 6- Step by Step DEBUG

- 25 Gravar login testes

- 1- Adicionar BlazeMeter Plugin no browser
- 2- Iniciar BlazeMeter plugin e login
- 3- Gravar o Cenário
- 4- Finalizar gravação e Exportar para .jmx
- 5- Importar .jmx em Jmeter
- 6- Adicionar os Listener

Seção 6: JMeter e Selenium - 26 Jmeter Selenium WebDriver

→ Analise de performance do cliente usando WebDriver

→ Add plugin

- JMeter > Plugin Manager > *Selenium/WebDriver Support*

- <https://jmeter-plugins.org/>

- <https://github.com/undera/jmeter-plugins-webdriver>

1- Criar um Test Plan/ Add Thread Group

2- Btn dir. Thread Group/ Add/ Config Element /jp@gc -Chrome Driver Config

3- ... / Add/ Sampler/jp@gc – WebDriver Sampler

4- ... / Add/ Listener/ View Results Tree

→ Download *chromedriver.exe* e definir o local no Chrome Driver Config

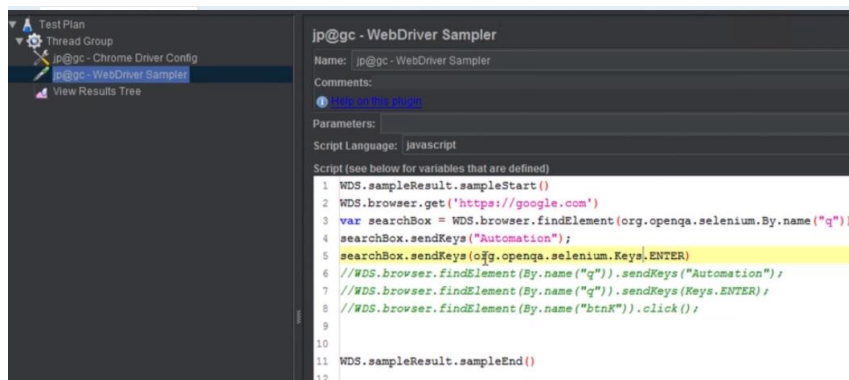
5- Adicionar Path ao *jp@gc Chrome Driver Config*

e. g. – D:\Desktop\drivers\chromedriver\chromedriver.exe

6- Adicionar script no Web Driver Sampler

7- Adicionar Extensão do Chrome **Katalon Recorder (Selenium IDE for Chrome)**

8- Adicionar um Script no Web Driver Sampler



→ **WebDriver Sampler** automatiza a execução e coleta das métricas da performance do navegador (cliente-side)

→ Enquanto o uso do WebDriver Sampler, cada Thread vai ter um único navegador instanciado e um consumo significativo entre as pesquisas

Seção 7: Dicas e Truques - 27 Encontrar LOGS no JMeter

1- Options > Log Viewer[V]

Options > Log Level > { ERROR/ WARN/ INFO/ DEBUG/ TRACE }

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

2- Jmeter/ bin/ log4js.xml

3- Escolher o Log Level e executar a aplicação JMeter

- 28 Passo a Passo Debug

1- Criar um Test Plan

2- Adicionar Server Name or IP ao HTTP Request

3- Plugin Manager → 3 Basic Graphs

- Step by Step Debugging

- <https://github.com/Blazemeter/jmeter-debugger>

- 29 Mudança do Timestamp formato em csv

1- Criar um Teste Plan

2- Adicionar Listener/ Aggregate Report

3- Adicionar um Filename no Write results para arquivos do Aggregate Report (results.csv)

4- Verificar results.csv

- 30 Uso de JSON Extractor

→ <https://regres.in>

→ adicionar o caminho [/api/users?page=2](https://regres.in/api/users?page=2) (<https://regres.in/api/users?page=2>)

→ Copiar resultado da página json

- 1- Criar um TestPlan
- 2- No HTTP Request adicionar
 - a. Protocol [http]
 - b. Server name or IP
 - c. Path
 - d. Method GET
- 3- Btn Dir. no HTTP Request/ Add/ Post Processors/ JSON Extractor
- 4- JSON - adicionar *Names of created variables*
- 5- JSON – Adicionar *JSON Path expressions* (EX: `$.id`; `$.first_name`)
- 6- Realizar o *Step by Step Debugger*

→ Dentro do Path pode-se adicionar um valor (ex: `/api/users/${User1}`)

- 31 Definir Threads/users por command line

- 1- Criar um Test Plan
- 2- Thread Group
 - *Number of Threads(users):* `$_P(User,1)`
 - *Ramp-Up Period (in seconds):* `$_P(rampUp,1)`

→ Help > Useful links > Functions Reference Documentation

<https://jmeter.apache.org/usermanual/Functions.html>

- 3- Salva o Test Plan
- 4- CMD:
`$jmeter -n -t "C:\Users\nomeUsuario\Desktop\JmeterTest\SeuTest.jmx"`
- 5- Options > Function Helper Dialog
 - Choose a Function
 - Definir um Name e Value
 - Gerar Function
- 6- CMD:
`$jmeter -n -t "C:\Users\... \JmeterTest\SeuTest.jmx" -JUser=5 -JRampUp=1`

- 32 Acrescentar registro de data e hora no arquivo de resultado
| Criar Nomes Exclusivos de Arquivos de Resultados

- 1- Criar TestPlan com HTTP Request
- 2- Adicionar Aggregate Report
- 3- Options > Function Helper Dialog
 - Choose a Function > __time
 - Format String: MM-dd-yyyy-HH-mm-ss
 - Name of variables: TimeVar
 - Gerar Function
 - `$_time(MM-dd-yyyy-HH-mm-ss,TimeVAR)`
- 4- Adicionar function ao filename do Aggregate Report com \\ ou /

Ex: ...\\jmeter\\test\\report\\`$_time(MM-dd-yyyy-HH-mm-ss,TimeVAR)`

- 33 Thread Groups Simultâneos – Configurar testes Reais
*** 15 Setup Performances Reais ***

→ jmeter-plugins.org/wiki/ConcurrencyThreadGroup

→ Download plugin

→ Plugins Manager

1. Add/ Threads/ bzm-Concurrency thread Group
2. Target Concurrency → Numero de campos que vão ser rodados paralelos
3. Ramp Up Time → Minutos ou Segundos
4. Ramp-Up Steps Count → partições do gráfico
5. Hold Target Rate Time(sec) → Momento de uniformidade
6. Thread Iterations Limit → Numero de vezes
7. Log Threads Status into file → arquivo de log

- 34 Criar Resultados Agregados csv por Command Line

1- Criar Thread Group + HTTP Request + Aggregate Report

→ `$ cd ../jmeter/bin`

→ `$ jmeter -n -t "C:\Users\Desktop\JmeterTest\jmeter_test.jmx" -l "C:\Desktop\JmeterTest\Report\result.jtl"`

→ JMeter Plugins CMD

→ <https://jmeter-plugins.org/wiki/JMeterPluginsCMD>

→ Download JMeterPlugins-Standard-1.4.0.zip

→ <https://jmeter-plugins.org/downloads/old>

→ Copiar todos os arquivos da pasta do arquivo baixado, JMeterPlugins-Standard-1.4.0.zip > lib > ext, e colar na pasta do próprio JMeter, JMeter > apache-jmeter > lib > ext

→ `$ cd ../apache-jmeter/lib/ext`

→ `$ JMeterPluginsCMD.bat --generate-csv "local do arquivo AggregateReport.csv" --input-jtl "local\do\arquivo result.jtl" --plugin-type AggregateReport`

-35 Como declarar a contagem de NODES de uma resposta JSON

- 1- Criar TestPlan com HTTP Response
- 2- Sample Rest API → reqres.in
- 3- Adiciona a Request → reqres.in/api/users?page=2
- 4- JSON Editor Online → jsoneditoronline.org
- 5- Colar o resultado do Request no JSON Editor
 - No HTTP Request adicionar informações
 - Protocol[http]:** https
 - Server Name or IP:** reqres.in
 - Method:** GET
 - Path:** /api/users?page=2
 - View Results Tree
 - Response data** = Resposta JSON
- 6- Btn Dir. HTTP Request/ Add/ Post Processors/ JSON Extractor
 - **Names of created variables:** UserCount
 - **JSON Path expressions:** \$.data[*]
 - **Match No.(0 for Ramdom):** -1
- 7- Step-by-Step Debugger
- 8- ../Assertions/Response Assertion
 - **JMeter Variable Name to use:** UserCount_matchNr
 - **Patterns to Test:** Adicionar valor para a resposta
- 9- ../ Post Processors/ jp@gc – JSON Path Extractor
 - Destination Variable Name:** UserCount
 - JSONPath Expression:** \$.data[*]
- 10- Desabilitar *JSON Extractor*

-36 Como deixar em loop através de arquivo CSV – enquanto Loop

- 1- Criar um TestPlan
- 2- Adicionar Server Name or IP e salvar
- 3- Criar uma pasta Data e um arquivo data.csv
- 4- Adicionar numero de loops ao Thread Group
- 5- ../Config Elements/CSV Data Set Config
- 6- **CSV Data Set Config** – Filename: Adicionar o caminho do arquivo data.csv
- 7- **HTTP Request** – Server name or IP: `${URL}`
- 8- Adicionar na coluna A do arquivo data.csv o valor URL e abaixo os endereços que deseja verificar
- 9- Btn dir. TG/ Add/ Logic Controller/ While Controller
 - **Condition (function or variable)** `${URL}`
- 10- Mover CSV Data Set Config e HTTP Request para dentro do While Controller
- 11- CSV Data Set Config
 - Ciclo no final do arquivo
 - **Recycle on EOF?**: False → Não manter uma trade ou parada súbita no final do arquivo
 - **Stop thread on EOF?**: True → O final do arquivo será alcançado e não manterá a continuação

-37 Time Functions

Function Helper

- a)- **Choose a function:** `__time`
 - **Format String:** MMM-dd-yyyy-HH-mm-ss
 - **Generate & Copy to clipboard**

- b)- **Choose a function:** `__timeShift`
 - **Format String:** MMM-dd-yyyy
 - **Date to shift:** Pode-se adicionar uma data ou fazer a não, caso não a

verificação sera feita com a date diaria

- **Amount:** P1D(pass 1 day) → verificar dia seguinte
P-1D(pass 1 day ago) → verificar dia anterior

-38 Executar um Thread específico por command line

- 1- Criar Vários(2 mínimos) Test Plan/ Thread Group /HTTP Request
- 2- Abrir o cmd
→ `$ jmeter -n -t "local do arquivo Threads.jmx" -l "local\do\arquivo result.csv"`
- 3- Function Helper
 - **Choose a function:** __P
 - **Name of property:** users
 - **Default value:** 1
 - `*${__P(users,1)}`
- 4- Retorna ao TG que deve ser testado especificamente
 - **Number of Threads(users):** `*${__P(users1,1)}`
- 5- No outro TG
 - **Number of Threads(users):** `*${__P(users2,1)}`
 - `$ jmeter -n -t "local do arquivo Threads.jmx" -l "local\do\arquivo result.csv" -Jusers1=2 -Jusers2=0`

-39 Recording in JMeter

- File < Templates < Recording
- File < Templates < Recording with Think Time
- HTTP(S) Test Script Recorder
 - Start
 - Gera um certificado no diretório bin para adicionar ao navegador
- BlazeMeter
- Adicionar um View Results

Seção 8: Server Health Check - 40 Monitorar Server Health | CPU, Memory, etc. durante teste

* 12 Plugin Manager

Response

- Time
- Avg Time
- Connect Time
- Size
- Bytes

Server – Máquina onde sua aplicação teste é hospedada, CPU, Memory, Disk I/O, Network

- Documentação PerfMon: jmeter-plugins.org/wiki/PerfMon

- 1- Adicionar PerfMon Plugin ao JMeter
- 2- Download e adicionar arquivos do plugin a pasta lib e ext
- 3- Habilitar no Plugin Manager

→ Server

- 4- Download Perform Server Agent

- <https://github.com/undera/perfmon-agent/blob/master/README.md>

- 5- Executar o arquivo *startAgent.bat*

- No Mac/Linux há necessidade de se executar pelo terminal `$sh startAgent.sh`

- Verificar se o Cliente tem comunicação com o Server via port 4444

- Abrir o cmd e localizar o IP `$ipconfig`

- Verificar Comunicação `$telnet 192.168.99.1 4444` → IP exemplo

- Ao iniciar o comando cmd, poderá checar a comunicação através do startAgent.bat executado



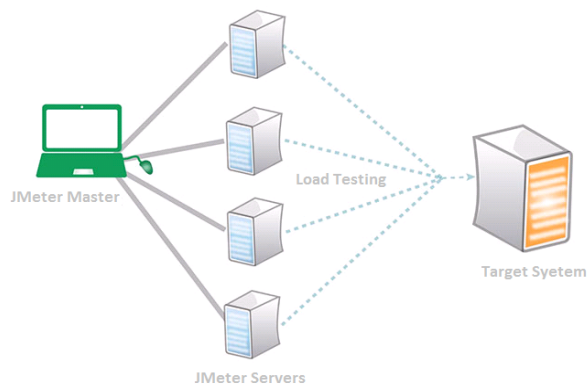
- Para finalizar comunicação basta digitar *shutdown* no cmd

- 6- Criar um Teste Plan no JMeter para verificar a saúde do Servidor

- 7- Btn TH/ Add/ Listener/ jp@gc – PerfMon Metrics Collector
 - ➔ **jp@gc – PerfMon Metrics Collector**
 - Servers to Monitor (ServerAgent must be started, see help)
 - Metric to Collect [CPU | Memory | Swap | TCP | Etc.]
- 8- Download Jenkins Generic Java package (.war)
 - Pelo CMD acessar a pasta Jenkins
 - `$java -jar jenkins.war --http Port=9191`
- 9- Criar um TG com HTTP Request simples
- 10- Executar o Test Plan
- 11- verificar resultados no jp@gc

Seção 9: Remote Testing | Master-Slaves

- 41 JMeter | Remote Testing | Master-Slave | Distributed Testing



- 1- Setup Master
 - adicionar um sistema ip remoto ao jmeter.properties
 - `apache-jmeter > bin > jmeter.properties.txt`
 - encontrar o remote_host IP
 - abre cmd
 - `$ipconfig`
 - pode-se adicionar múltiplos IP Address separados por virgula
 - `remote_host=192.168.1.2,192.168.1.3`

2- Criar um arquivo keystore

- apache-jmeter > bin > **create-rmi-keystore.bat**
- abrir no cmd

```
$ cd jmeter/bin
$ ./create-rmi-keystore.sh
What is your first and last name?
[Unknown]: rmi
What is the name of your organizational unit?
[Unknown]: My unit name
What is the name of your organization?
[Unknown]: My organisation name
What is the name of your City or Locality?
[Unknown]: Your City
What is the name of your State or Province?
[Unknown]: Your State
What is the two-letter country code for this unit?
[Unknown]: XY
Is CN=rmi, OU=My unit name, O=My organisation name, L=Your City, ST=Your State, C=XY correct?
[no]: yes

Copy the generated rmi_keystore.jks to jmeter/bin folder or reference it in property 'server.rmi.ss
```

- <http://jmeter.apache.org/usermanual/remote-test.html>
- name: *rmi*
- password: {definir}

3- Run jmeter-server file on slave (máster) system

- Criar um Test Plan
- Run > Remote Start > 192.168.1.2

4- Executar jmeter-server.bat

5- Executando pelo cmd

- `$jmeter -n -t "Users/user/View results tree.jmx" -l "users/user/result.csv" -R 192.168.1.2`

- Todos sistemas (Master e Slaves) tem mesma versão do JMeter
- Todos sistemas têm Java (ideal mesma versão)
- Todos sistemas podem conectar um com o outro (mesma subnet)
 - subnet = máscara de rede
- Não precisa copiar scripts JMeter(jmx) para sistemas slaves
- Se necessitar de 100 users e usando 2 slaves. Não passar como 50.

Seção 10: JMeter on Linux - JMeter on Amazona ws Linux ec2

- Checar se o Java está instalado
- Abrir JMeter
- Extrair JMeter (tar -xf apache-jmeter-4.0.tgz)

1- aws - Connect to Your Instance

2- IOS/Linux :

```
$ssh -i "Linux_April2017.pem" ec2-user@ec2-34-237-142-2002.compute-1.amazonaws.com  
$ ./jmeter.sh -n -t examples/
```

Seção 11: New Versions - JMeter 4.0 | NEW

- Neat and clean looking.
- Look and Feel > ...
- Support Java 9
- Sem Workbench → Workbench = HTTP(S) Test Script Recorder
- Help > Useful Links | Export Transactions for reports
- Save Your test plan whenever run it
- JMeter emulates a group of users sending request to a server
- Collects response and Metrics of the communication
- Presents the results in various formats via tables and graphs

Fontes Extras :

- <http://shipit.resultadosdigitais.com.br/blog/testes-de-carga-com-jmeter/>
- <http://www.decom.ufop.br/imobilis/metodologia-de-testes-tutorial-jmeter-para-testes-de-performance-em-plataforma-web/>
- <http://jmeter.apache.org/usermanual/build-db-test-plan.html>
- <https://jmeter.apache.org/usermanual/generating-dashboard.html>
- <https://jmeter-plugins.org/wiki/PluginsManager/>
- <http://www.decom.ufop.br/imobilis/metodologia-de-testes-tutorial-jmeter-para-testes-de-performance-em-plataforma-web/>
- <https://stackoverflow.com/questions/48780603/workbench-section-not-showing-in-apache-jmeter>
- <https://www.devmedia.com.br/teste-de-performance-com-jmeter/34621>
- <https://reqres.in/>