https://blog.csdn.net/ronghuanye/article/details/72956054

**jmeter使用教程**

2017年06月09日 11:40:38 [ronghuanye](https://me.csdn.net/ronghuanye" \t "_blank) 阅读数：58379更多

个人分类： [性能测试](https://blog.csdn.net/ronghuanye/article/category/6963645)

**一、安装Jmeter**

**1、下载Jmeter、JDK、badboy**

**Jmeter官网下载地址**：

<http://jmeter.apache.org/download_jmeter.cgi>

**JDK官网下载地址**：

<http://www.oracle.com/technetwork/java/javase/downloads/java-archive-downloads-javase7-521261.html>

**Badboy官网下载地址**：

<http://www.badboy.com.au/download/add>

**已下载的安装包，在18跳板机E:\软件**

**2、安装JDK**

安装Jmeter之前需要先配置Java环境，我们下载的是jmeter3.0，所以java版本需要java7以上的版本。

安装JDk的步骤如下：

点击下载的JDK文件jdk-7u13-windows-i586.exe，点击下一步直至安装完成，然后开始配置环境：

点击我的电脑----属性----高级----环境变量----在系统变量中----点击新建，

在变量名中输入：JAVA\_HOME

变量值中输入：C:\ProgramFiles\Java\jdk1.7.0\_13

再次点击新建：

在变量名中输入：CLASSPATH，

变量值中输入：.;%JAVA\_HOME%/lib/dt.jar;%JAVA\_HOME%/lib/tools.jar;

然后再系统变量中找到Path，点击编辑，在变量值中加上：

%JAVA\_HOME%/bin;%JAVA\_HOME%/jre/bin;（如果前面没有“;”需要加上）

配置完成之后，点击确定保存，然后cmd打开命令窗口输入：java或者javac，出现大串字符串表示配置成功。

也可以输入：java-version 查看java版本来查看是否配置成功

OK，JDK安装成功之后，就可以下一步安装Jmeter了。

LINUX下安装JDK请参考相关文档进行安装。

**3、安装Jmeter**

安装jmeter很简单，只要将下载的zip包解压缩到指定文件夹下即可，例如将zip包解压缩到E盘根目录下面：E:\apache-jmeter-3.0

安装好后的目录结构如下：

E:\apache-jmeter-3.0\bin

E:\apache-jmeter-3.0\docs

E:\apache-jmeter-3.0\extras

E:\apache-jmeter-3.0\lib

E:\apache-jmeter-3.0\lib\ext

E:\apache-jmeter-3.0\lib\junit

E:\apache-jmeter-3.0\licenses

E:\apache-jmeter-3.0\printable\_docs

如果你想的话你可以重命名父目录（例如jmeter-3），但是不要改变任何子目录命名。父目录路径中不能包含任何空格；如果包含，你运行客户端-服务器模式会有问题。

**4、安装badboy**

安装bayboy也很简单，只要双击exe安装文件，然后一路next就可以了。

安装完成后在开始菜单中点击badboy菜单即可启动

**二、准备测试脚本**

**1、录制Jmeter脚本**

录制Jmeter脚本有两种方法，一种是设置代理；一种则是利用badboy软件，badboy软件支持导出jmx脚本。

这里我们介绍第二种方法，利用badboy录制脚本，然后导出Jmeter需要的jmx文件。

打开badboy时默认是在录制模式下，可以在窗口标题栏中看到（recording），如果点击工具栏中个红色圆形按钮则进入非录制模式。首先，在地址栏中输入要录制脚本的地址，然后点击绿色按钮即可开始录制；录制完成后点击红色圆形按钮旁边的正方形黑色按钮结束脚本录制（录制的过程和loadrunner类似），在内嵌的浏览器上面操作你需要测试的动作即可，它会记录你的相关行为动作。

录制完成后点击黑色按钮完成录制。

导出jmx文件：

保存为“login.jmx”，

**2、打开脚本**

使用jmeter打开上面保存的文件“login.jmx”

打开之后如图所示：

**3、修改脚本**

**a、线程组**

如果是录制方式打开的jmx脚本默认会场景线程组，如果是手工生成脚本则需要首先添加线程组，右键点击测试计划,选择添加,添加线程组，这里的线程组相当于loadrunner中的用户组。

如上图：

名称：可以随意取，最好取为我们测试场景相关的名称

注释：随意

在取样器错误后要执行的动作：我们选择继续，错误之后依然继续执行

线程数：一般我们用来表示多少个用户，相当于loadrunner中的并发用户数

Ramp-up Period(in Seconds)：表示每个用户启动的延迟时间，上述我设为1秒，表示系统将在1秒结束前启动我设置的1000个用户，如果设置为1000秒，那么系统将会在1000秒结束前启动这1000个用户，开始用户的延迟为1秒，如果我设置为0秒，则表示立即启动所有用户。相当于loadrunner中的用户加载策略。

循环次数：如果你要限定循环次数为10次的话，可以取消永远的那个勾，然后在后面的文本框里面填写10；在这里我们勾上永远，表示如果不停止或者限定时间将会一直执行下去，是为了方便调度器的调用。

调度器的配置：我们勾选调度器时，将会出现这个面板

启动时间：表示我们脚本开始启动的时间，当你不想立即启动脚本测试，但是启动脚本的时间不会再电脑旁的时候，你可以设定一个启动的时间，然后再运行那里点击启动，系统将不会立即运行，而是会等到你填写的时间才开始运行。

结束时间：与启动时间对应，表示脚本结束运行的时间。

持续时间：表示脚本持续运行的时间，以秒为单位，比如如果你要让用户持续不断登录1个小时，你可以在文本框中填写3600。如果在1小时以内，结束时间已经到达，它将会覆盖结束时间，继续执行。

启动延迟：表示脚本延迟启动的时间，在点击启动后，如果启动时间已经到达，但是还没有到启动延迟的时间，那么，启动延迟将会覆盖启动时间，等到启动延迟的时间到达后，再运行系统。

注意：如果我们需要用到调度器来设定持续时间，如果线程数不够多到持续时间结束，我们就必须将循环次数勾选为永远，特别地，如果线程组里面有其他的循环，我们也需将该循环次数勾选为永远（如我上面录制的脚本中的Step1也是一个循环，需要将永远勾选），否则，按我如上配置，将永远去掉勾选，文本里填1，那么无论你将持续时间启动时间结束时间等设置多少，系统运行1000次后，将会停止不再运行。

**b、关于HTTP请求**

关于HTTP请求，我们录制的脚本里面就有，这里就介绍一些常用的属性。

如图所示：

其中：

名称：可以随意取，我们可以根据录制脚本的路径来命名

注释：可以对该界面做一个简单介绍

服务器名称或IP：即我们的服务器或者IP的地址。

端口号：我用的是18080端口，所以设置的端口号为18080

后面的超时定义可以不用填写。

Implementation：这里我们录制时默认填写为JAVA

协议填写为：http

方法为：GET

Content encoding：编码可以不用填写

路径：即页面的路径；下面的重定向等选择可以根据需要勾选，这里我们保持录制不修改。

Parameters参数：即跟着路径一起发送的参数及文件

下面的代理服务器等可以忽略，按需要填写。

**c、参数化**

如下面图，我们为请求发送的参数来参数化。

上述的例子以登录为例，登录有两个参数：userAccount和password，在实际测试中，当我们需要用到不同的用户来登录的时候，我们就可以对userAccount和password参数化。

1)、我们新建一个文件，这里我新建的是一个txt文件，命名为c.txt，里面的内容为

300000000@qq.com,1qaz!QAZ

300000001@qq.com,1qaz!QAZ

300000002@qq.com,1qaz!QAZ

2)、在jmeter中的【选项】中选择【函数助手对话框】，将会弹出如下对话框：

其中：

CSV file to get values from |\*alias：要读取的文件路径，为绝对路径

CSV文件列号| next| \*alias：从第几列开始读取，注意第一列是0

如上图所示，我们读取的是c.txt里面的第一列用户名(如果要读取第二列的密码，只需将0改成1即可,往后类推)，点击【生成】按钮即可生成函数，我们使用时即拷贝生成的函数字符串：${\_\_CSVRead(E:\c.txt,0)}。

**d、关联**

Jmeter中的关联是使用正则表达式提取器来实现的，相对于loadrunner来说jmeter的关联功能比较弱，只能在线程组内实现关联，不能跨线程组。

我们以进入vmall商品列表，然后打开某个上面详情页为例进行关联操作：

第一步：右键点击商品列表的请求> 添加> 后置处理器 >正则表达式提取器

第二步：填写正则表达式

我要获取的值原本服务器返回的结果是：以下是左边界和右边界

解释：  
（1）注释不多说，随便自己喜欢  
（2）Apply to:默认即可  
 (3) 要检查的字段：一般我们选择主体，即服务器返回给我们的页面主体信息  
（4）引用名称：即参数名称，这个自己定义，在后面时可用  
（5）正则表达式： 正则表达式中()括起来的部分就是要提取的。.代表任意字符，+代表出现任意次，后面加？。  
（6）模板：$1$代表只取一组数据  
（7）匹配数字：0代表随机取值，1代表取找到的第一个值，2代表去找到的第二个值，依次类推。  
（8）缺省值：如果参数没有取得到值，那默认给一个值让它取

第三步：在商品详情页请求中使用关联的参数值

在第二步填写的引用名称中加入${}，然后替换掉需要关联的地方，这里把整个url替换了。

**e、断言验证**

在web测试中，有的时候，即使我们测试返回的response code为200，也不能保证该测试是正确的，这时候，我们可以用到响应断言，通过对比响应的内容来判断返回的页面是否是我们确定要返回的页面，或者判断请求是否成功，相当于loadrunner中的检查点。

选择我们要判断的页面，点击添加断言----响应断言，将会出现如下图：

名称：随意，这里我们不做改动

注释：随意

Apply to：这里我们选择默认值

要测试的响应字段：这里我们选择响应的文本，即返回的页面信息

模式匹配规则：这里我们选择包括，即响应的文本是否包括我们验证的信息

要测试的模式：title="商品详情">，即我们需要验证， title="商品详情"> 是否在响应文本中存在，也可以添加更多的字段来验证。

如果没有找到需要匹配的字符串则事务会失败。

**4、调试脚本**

调试脚本一般是通过添加查看结果树监听器、查看jmeter运行日志、查看控制台日志实现

查看结果树

Jmeter运行日志

控制台日志

有时jmeter的控制台也会打印错误日志，如果是java请求打印的显示结果会在这里显示。

**5、添加setUp ThreadGroup、tearDown Thread Group线程组**

setUp Thread Group 相当于loadrunner的vuser\_init部分，在运行过程中只执行一次，用于初始化动作。

tearDown Thread Group 相当于loadrunner的vuser\_end部分，在运行过程中只执行一次，用于关闭动作。

根据测试场景是否需要来添加setUpThread Group、tearDownThread Group线程组。

**三、添加监控**

Jmeter的监控图表分为测试结果监控图表和服务器资源监控图表

**a、添加测试结果监控**

测试结果监控主要是通过添加各种监听器实现的，添加监听器之前先安装一个插件

JMeterPlugins

**JMeterPlugins的下载地址**

地址：[http://code.google.com/p/jmeter-plugins/downloads/list](http://code.google.com/p/jmeter-plugins/downloads/list" \t "_blank)，选择需要的版本，下载

JMeterPlugins的配置使用

将下载的文件解压，得到如下目录：

**已下载的安装包，在18跳板机E:\软件**

将JMeterPlugins.jar包复制到Jmeter的lib目录下面的ext目录下面,重新启动Jmeter，我们点击添加就可以看到出现了很多的jp@gc-开头的监听器。

一般我们要添加的监听器有聚合报告、活动线程变化图、响应时间图、TPS、组合图，如果需要看到断言信息也可以选择断言结果。

**b、添加服务器监控**

一般我们监控服务器的资源使用zabbix就行了，也可以使用插件提供的资源监控器进行监控。添加监控前需要将下载的插件目录下的serverAgent目录及下面的文件复制到我们要监控的服务器上，然后点击打开/serverAgent/startAgent.bat这个文件，它的默认端口为4444。（如果是linux系统则运行/serverAgent/startAgent.sh即可）

然后添加监听器jp@gc -PerfMon Metrics Collector，并且在服务器监控列表中添加需要监控的ip地址、端口、监控计数器，如下图：

当运行测试场景时就可以看到监控的资源使用情况了。

**四、执行测试**

在修改好测试脚本、添加监控、设置好运行策略之后就可以执行测试了，一般情况下，如果是运行少量的并发用户可以使用单节点图形界面进行测试，如果是运行大量并发用户建议使用非GUI方式和集群方式进行测试。

**1、图形界面运行测试**

**2、非GUI方式运行测试**

在windows中打开cmd命令行界面，进入jmeter的bin目录下，预先把jmx脚本拷贝到bin目录下，然后执行命令jmeter-n.cmd login.jmx 执行界面如下：

执行完成后会在bin目录下面生成测试结果文件和运行日志文件

其中jtl文件是结果文件，可以在图形界面下导入到图表中

在linux中执行jmeter.sh  -n -t login.jmx -l login.jtl

**3、集群方式执行测试**

**Jmeter集群模型**

分为控制端和服务器端，控制端一般不产生测试压力，只负责控制服务器端的运行和收集测试结果并实时用图表的方式展现给用户，实际产生测试压力的是jmeter服务器端

**在windows下配置集群方式**

在jmeter的bin目录下找到jmeter.properties文件，找到remote\_hosts字段并在后面添加要加入集群的机器ip地址，中用逗号隔开

添加完成后在每台集群的机器上启动服务器端，执行bin目录中的jmeter-server.bat文件（如果是linux服务器则执行jmeter-server即可，执行前先给文件赋执行权限），然后会弹出命令窗口如下，不能关闭它。

重启jmeter后可以在运行菜单>远程启动下面看到已经添加的ip地址

点击对应的ip地址就会使用对应的机器执行脚本，或者点击远程全部启动则使用所有的机器执行脚本；也可以点击工具栏上的远程全部启动按钮启动测试。

使用集群方式执行测试过程中，每台测试机的测试结果将实时传回到控制台并合并显示在图表中。

或者在cmd命令窗口中使用非GUI方式启动集群执行测试

在windows中打开cmd命令行界面，进入jmeter的bin目录下，预先把jmx脚本拷贝到bin目录下，然后执行命令jmeter-n-r.cmd login.jmx 执行界面如下：

**在linux下配置集群方式**

在linxu下不需要修改配置文件jmeter.properties，只需要在每台集群的机器上启动服务器端，执行bin目录中的jmeter-server.bat文件（如果是linux服务器则执行jmeter-server即可，执行前先给文件赋执行权限）即可，然后执行下面命令：

jmeter.sh  -n -t login.jmx -l login.jtl -R192.168.1.1,192.168.1.2

启动参数说明：

-n 表示已非GUI方式启动

-t 后面是测试脚本文件路径

-l 后面是测试结果记录文件路径

-R 后面试远程jmeter服务器端ip地址，多个ip用逗号隔开

**五、分析结果**

**1、查看结果树**

如图所示：成功的为绿色，失败则显示为红色。如果测试的结果太多，你只需要看到错误的页面，则勾选【仅日志错误】

**2、聚合报告(AggregateReport)**

其中：

Label：标签，即我们上面的请求名称

#Samples：本次场景中一共发出了多少个请求

Average：平均响应时间

Median：中位数，也就是50%的用户的响应时间

90%Line：表示90%的用户的响应时间，如果最小值和最大值相差很大的话，我们一般选择这个作为最终测试结果

Min：最小响应时间

Max：最大响应时间

Error%：出错率，本次测试中出现错误的请求的数量/请求的总数

Throughput：吞吐量

KB/sec：每秒从服务器端接受到的数据量

**3、响应时间**

**4、TPS**

**5、组合图表（将线程数和TPS或者响应时间组合起来看）**

**五、常见问题**

**1、linux下启动jmeter服务端报错**

如果启动过程中报错java.rmi.RemoteException:Cannot start. host-192-168-1-86 is a loopback address

**解决办法**：

（1）、修改/etc/hosts文件中IP地址和主机名的映射关系

修改前

修改后

修改后启动正常

（2）、在执行jmeter-server脚本后面加入参数-Djava.rmi.server.hostname=192.168.1.86

./jmeter-server -Djava.rmi.server.hostname=192.168.1.86&

说明：192.168.1.86为本机IP地址

**2、查看结果树没有禁用导致测试结果不准确**

在调试脚本的时候可以使用查看结果树，方便调试，但是在执行测试过程中需要将查看结果树删除或者禁用。

**3、在进行java的rpc接口调试时报fastJSON解析异常错误**

com.alibaba.dubbo.rpc.RpcException: Failedto invoke the method queryDisplayProduct in the service pms.rpc.PmsInterface.Tried 2 times of the providers [192.168.1.86:20881] (1/1) from the registry192.168.1.54:2181 on the consumer 192.168.1.63 using the dubbo version 2.5.3.Last error is: Failed to invoke remote method: queryDisplayProduct, provider:dubbo://192.168.1.86:20881/pms.rpc.PmsInterface?anyhost=true&app.params={"name":"pms"}&application=pms\_consumer&check=false&dubbo=2.5.3&interface=pms.rpc.PmsInterface&methods=queryCarrier,queryDisplayProduct,querySbomByDisPrdIds,queryCategoryIdCode,querySbomByCodes,querySbomBaseInfo,queryAttributeValue,queryCategory,queryExtend,querySbomGift,querySbomBlobs,querySbomIdCode,querySbomBundle,querySbomList&pid=10136&retries=1&revision=0.0.2-SNAPSHOT&router=vmall&serialization=fastjson&service.filter=vmall&side=consumer&timestamp=1460450188935&tm=1460447837,cause: com.alibaba.fastjson.JSONException: not match : - "

com.alibaba.fastjson.JSONException: notmatch : - "

**解决办法**：把java工程的字符集改为UTF-8;在jmeter的启动文件jmeter和jmeter.bat的ARGS参数后面加上-Dfile.encoding=UTF-8

**4、大量节点集群执行测试，测试工具存在瓶颈，导致测试结果失真**

在进行dubbo多provider节点测试时，jmeter也使用多节点集群方式产生测试压力，如10个provider集群，则使用10个jmeter集群进行测试。当provider节点数从1增加到10个时，jmeter节点数也1增加到10，但是集群的TPS没有线性增长，同时响应时间反而下降，经过分析是测试工具JMETER多负载发生器本身存在性能瓶颈造成的，因为当jmeter服务端节点越多时，实时传送测试结果到控制端的数据就越多，使控制端产生很大压力，同时也会给jmeter服务端产生阻塞。

**解决办法**：不使用jmeter集群产生测试压力，使用多个jmeter独立运行产生测试压力。

**5、在进行java脚本开发时，java代码的执行效率影响测试结果**

一般需要自己编写的测试代码尽量要少，并且要简洁高效，否则可能使脚本存在性能问题而影响了测试结果。

**例如**：（1）下面是设置成功与否标志的两种写法，绿色框中的写法比红色框中的写法高效，测试出来的TPS会高一点。

（2）下面判断是否成功的变量success定义为全局变量，且初始化为true，每次循环时没有重新初始化，当发生错误时改变success的值为false，导致后面的循环都是使用false的值，并判断事务为失败状态。

解决办法：在每次循环开始是将success的值重新初始化为true。