**Jmeter sprint**

**Resource:**

1>

<http://newtours.demoaut.com>

use: mercury

pss: mercury

Jmeter做压力测试的范围远远不止于对Web，它还可以用于Java小服务程序、CGI 脚本、Java 对象、数据库， FTP 服务器, jdbc测试, XML等等

3> blazemeter – moble test/record for jmeter/plugin in chrome

4> 建议，在使用JMeter时，JDK的环境为1.6.

# User Case Sprint

1. Sprint1

-----通过JMeter自身设置来录制脚本

-打开JMeter工具

-创建一个线程组（右键点击“测试计划”--->“添加”---->“线程组 Thread Group”）

Or ---->“线程组 Thread Group”）->逻辑控制器->录制控制器。??(not need)

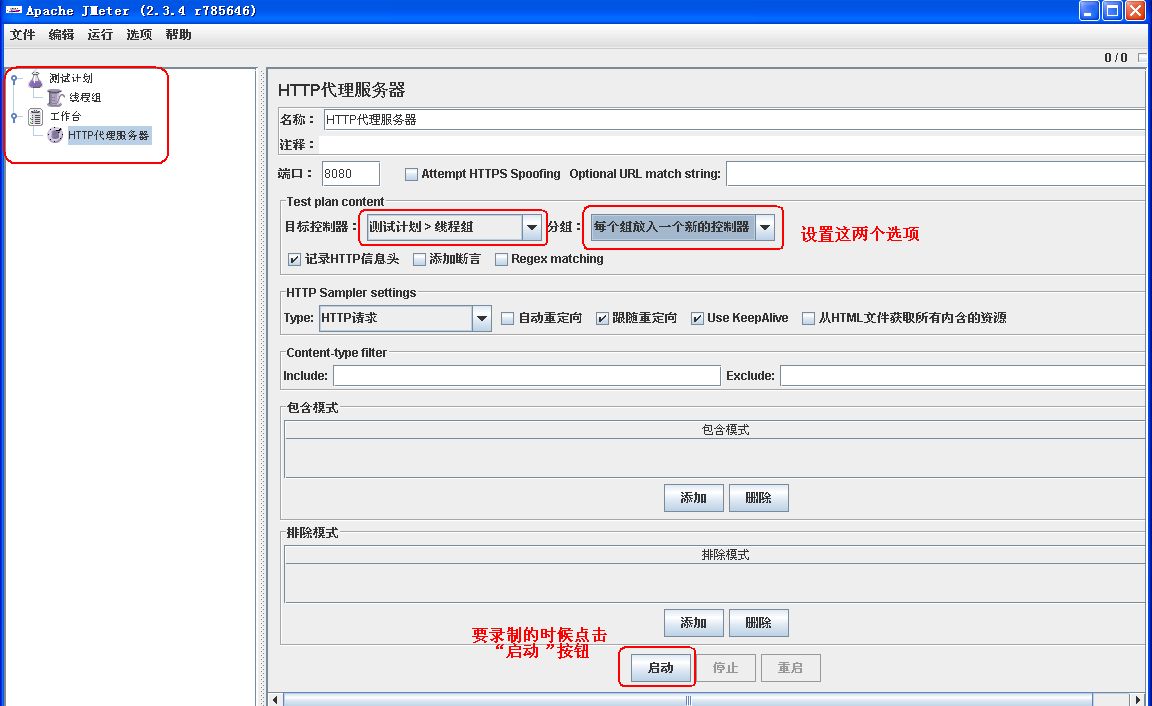
-创建一个http代理服务器（右键点击“工作台WorkBench”--->“添加”--->“非测试元件 non-test elements”--->“http代理服务器”

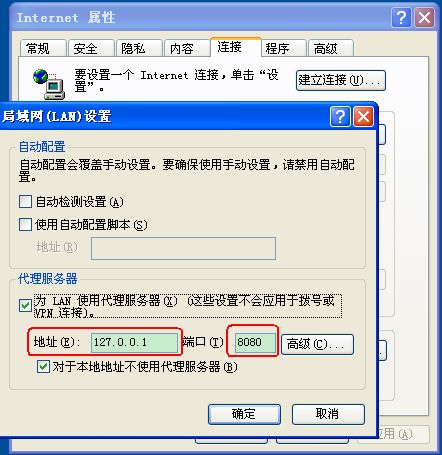
---->Test plan content

🡪> Target controller-(Testplan>thread group)

🡪>Grouping –(Put each group in a new controller)

🡪>Start



- IE--->“internet属性”--->“连接”--->“局域网设置” 

-- 输入需要测试页面的URL，在打开的页面进行操作，JMeter即会将所进行的操作录制成脚本；可以看到在“线程组”下看到新加的一系列节点。

--操作完成后，点击“停止”按钮停止录制，然后把浏览器的设置还原。

 --完成脚本的录制后还需要定义你所需的结果设置才算完成了Test Plan的设置，首先要设置Thread Group（相当于虚拟用户组）的属性，单击Thread Group，在右边的窗口中设置Number of Thread（相当于虚拟用户数）为5，Ramp-Up Period(in seconds)为1，Loop Cout选项取消选中Forever，在右边的框中输入2。以上设置表示共有5个虚拟用户，每隔1秒运行一个，迭代2次。  
 --要查看运行的结 果还需要加入Listener，选中Thread Group单击鼠标右键选择Listener->View Results in Table，通过这个视图可以查看到每个链接的执行情况，包括执行时间和成功与否，再加入Aggregate Report，通过这个视图可以查看每个Thread Group组的最大、最小和平均响应时间。加入View Results Tree，通过这个视图可以查看每个链接的执行情况，包括发出的请求和服务器返回的详细内容。  
**--运行及结果查看** (1)OK，运行看看吧，由于设置运行时间较短，所以先选择View Results in Table，然后按下Ctrl+r键开始运行脚本，就可以看到每个请求的运行结果了，还可以单击**[其他](javascript:;" \t "_self)**两个结果视图实时查看运行结果。

------[用BlazeMeter录制JMeter测试脚本](http://www.cnblogs.com/yangxia-test/p/5354111.html)

工具:

1，JMeter

2，Chrome

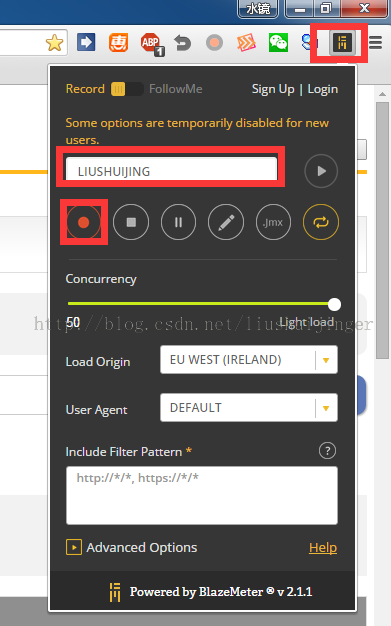
3，[BlazeMeter](http://files.cnblogs.com/files/yangxia-test/www.cnplugins.com_mbopgmdnpcbohhpnfglgohlbhfongabi_2_2_0_3_.zip" \t "_blank)

4，SwitchyOmega(如果需要代理)

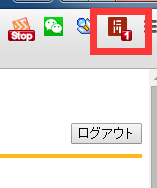
步骤:

以上工具准备好以后就可以录制JMeter的测试脚本了，

在Chrome中点击BlazeMeter插件,出现下图：

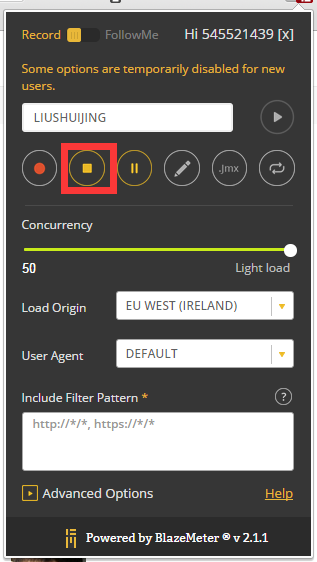


可以为本次录制取一个名字，然后点击红色原点,开始录制脚本，然后开始录制你需要测试的功能。

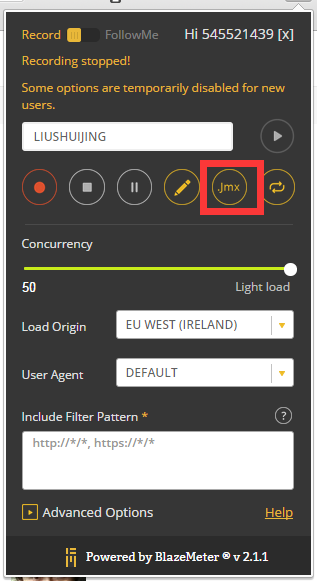


每一次服务器请求，BlazeMeter插件的图标上的数字都会加一，可以清楚的看到你录制的步骤。

当录制完成后，再次点击BlazeMeter的图标，然后点击黄色的停止按钮：

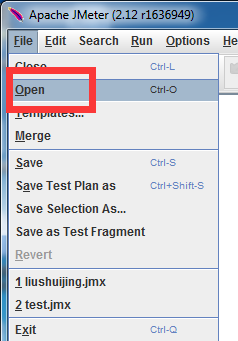


接着点击那个“jmx”按钮，将测试脚本导出：

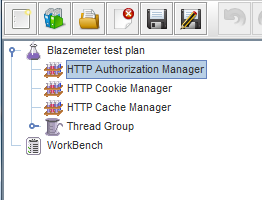


最后会生成一个.jmx的文件，自己取个名字保存到相应的位置即可。

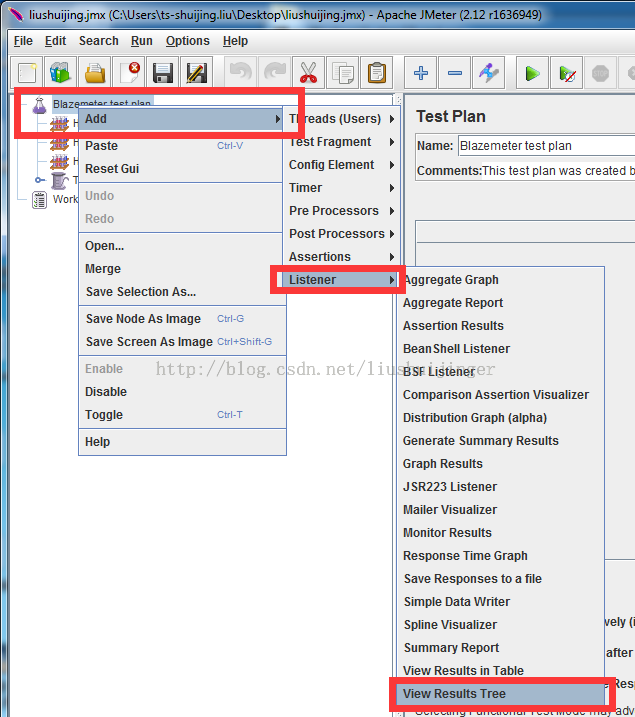
打开Jmeter，file—>open：找到刚才保存的脚本文件，打开：



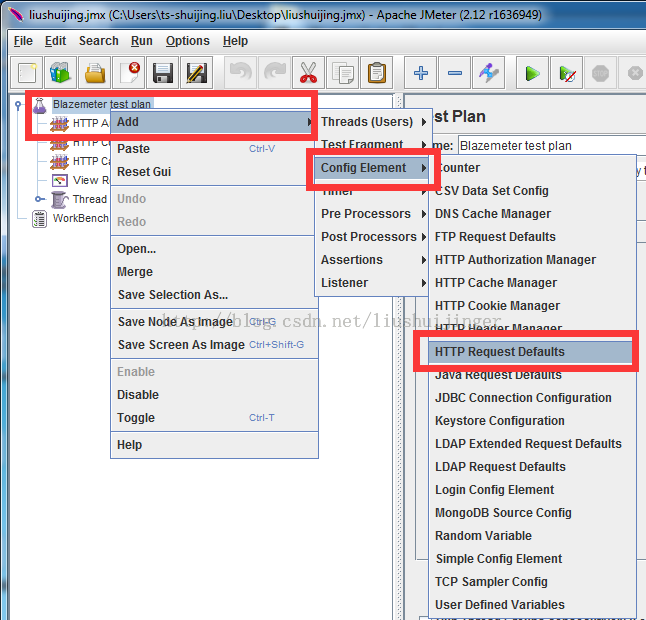
打开以后看到如下界面说明脚本已经导入了：



然后添加一个“view result tree”以便查看测试的详细信息：



如果需要代理的话还需要建一个“http request default”：

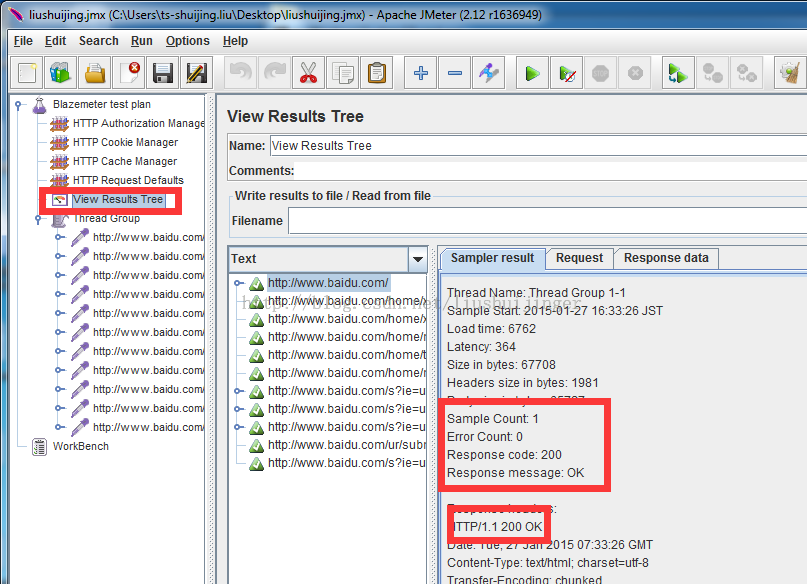


然后在其属性页面填入相应的代理信息即可（这个是全局代理，每个请求还可以单独设置自己的代理）。

然后点击工具栏中的绿色的开始按钮：

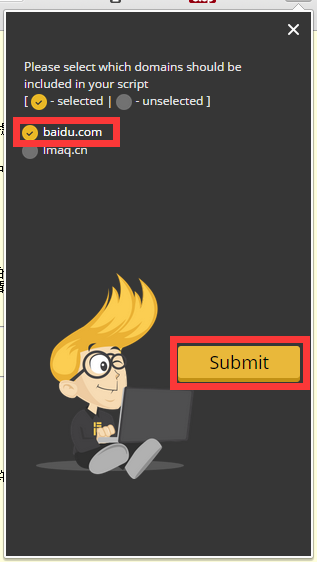


切换到“view result tree”视图，观察测试结果：



由上图我们可以看到请求的错误数、返回的状态码等信息，最右侧还可以切换不同的标签，来查看不同的返回信息。

PS：测试的时候浏览器最好只打开一个标签，以免其他页面对录制的脚本造成影响，如果涉及到多个站点的请求，在导出脚本的时候会让你选择：



1. Sprint2

Build A JMeter Web Test Plan: (5 users) x (2 requests) x (repeat 2 times) = 20 HTTP requests

(1). Add User (add the ThreadGroup): click Test Plan, mouse right click add button to get the Add menu, and then select Add --> ThreadGroup.

(the number of users you want to simulate: 5, how often the users should send requests: delay 1, and the how many requests they should send 2.)

(2). Add HTTP Request Defaults: selecting the JMeter Users (Thread Group) element. Click your right mouse button to get the Add menu, and then select Add --> Config Element --> HTTP Request Defaults Enter IP: jmeter.apache.org

(3). add HTTP Request: JMeter Users element (Thread Group) (Add --> Sampler --> HTTP Request). Then, select the HTTP Request element in the tree and edit the following properties

Following properties

.Change the Name field to "Home Page".

.Set the Path field to "/".

(4), add another HTTP Request: (Same as 3 step) .

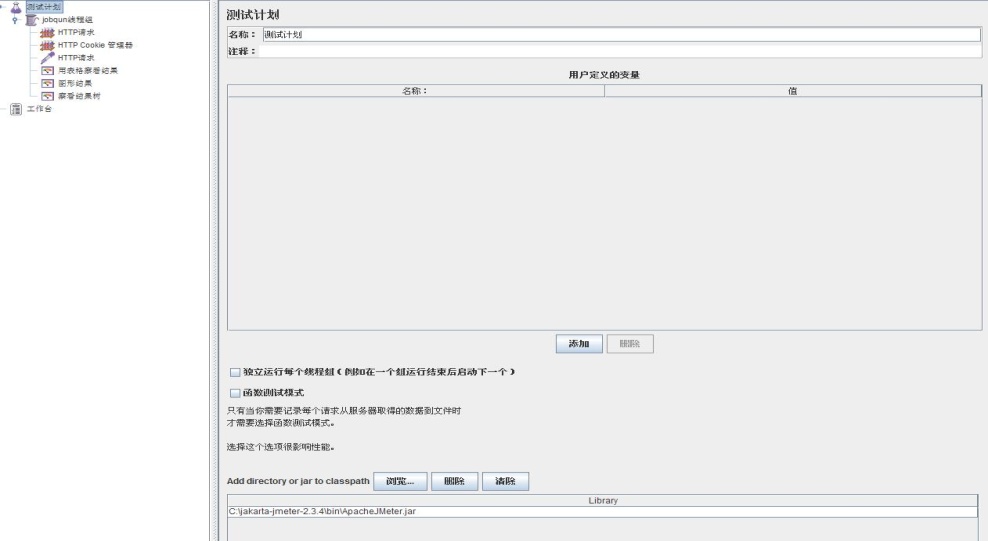
.Change the Name field to "Changes".

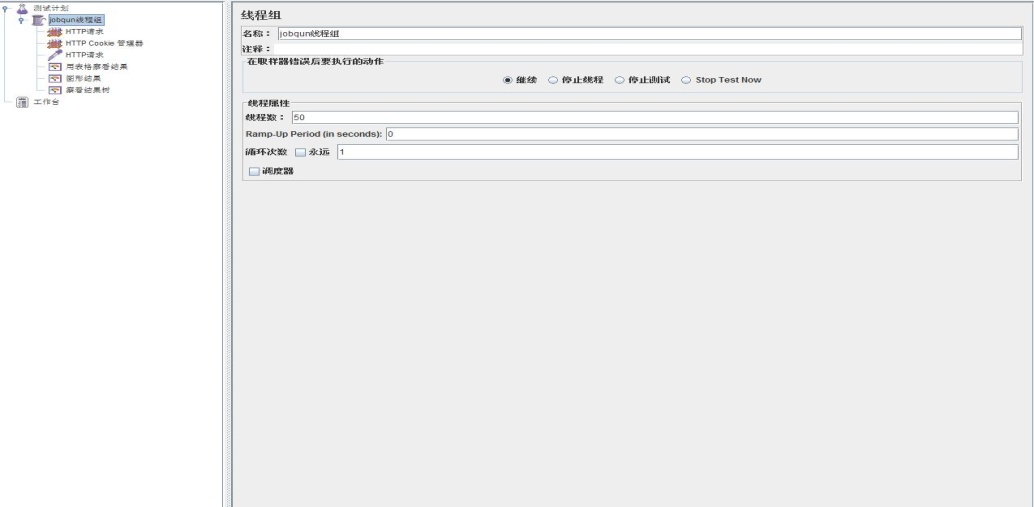
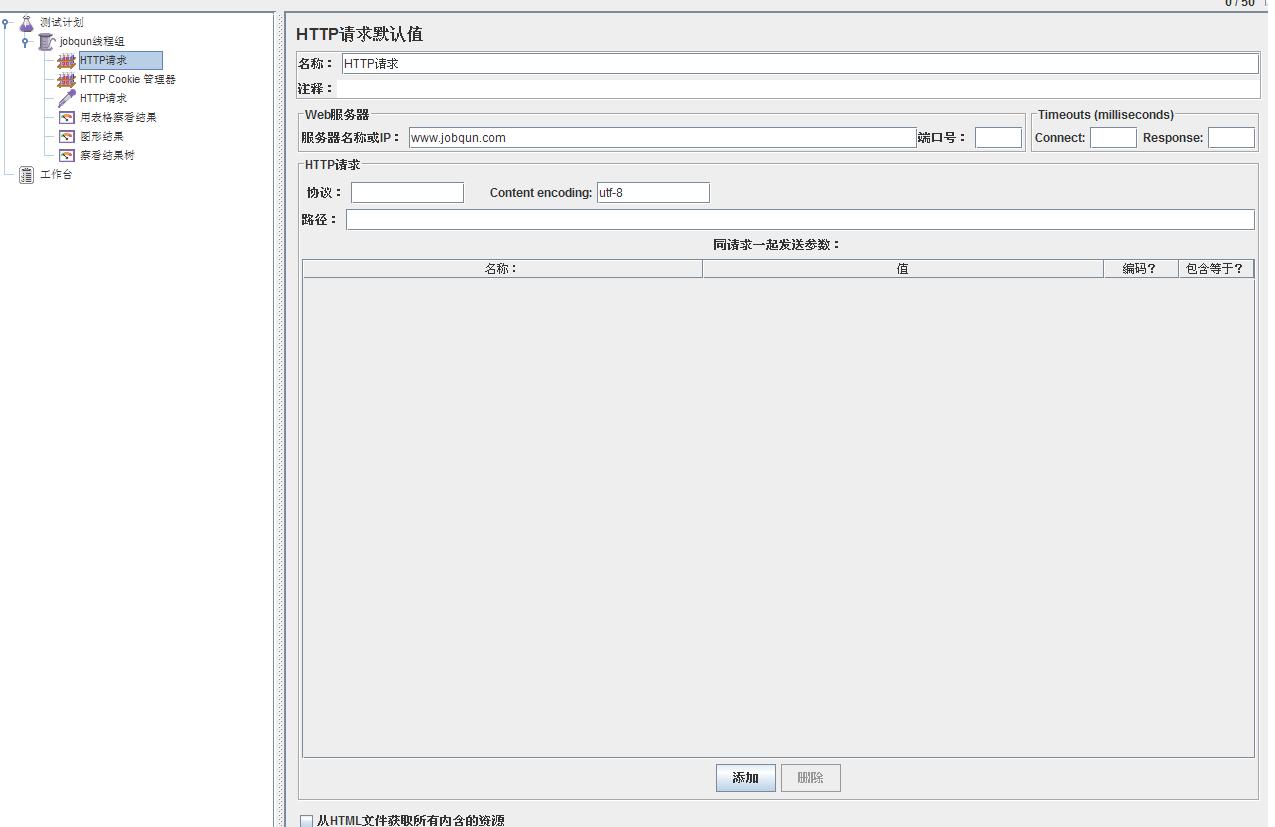
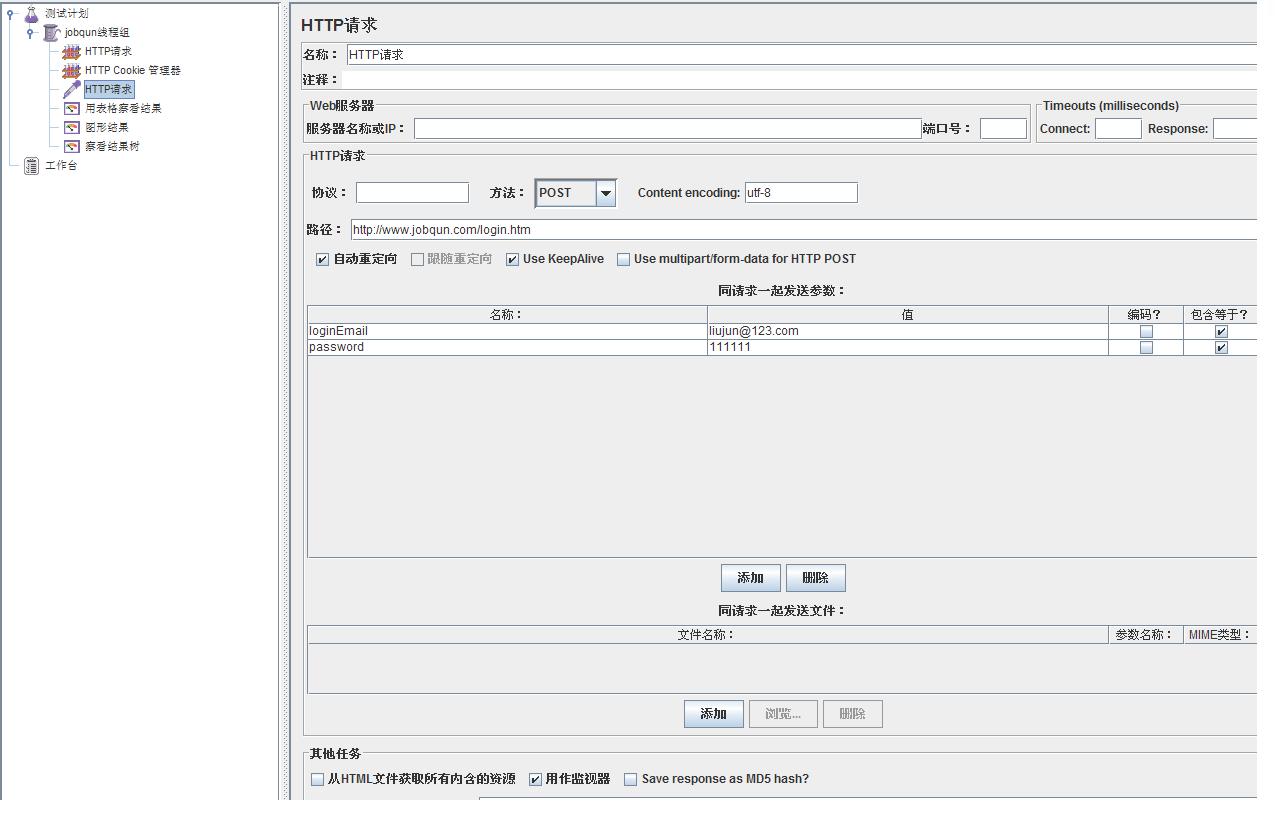
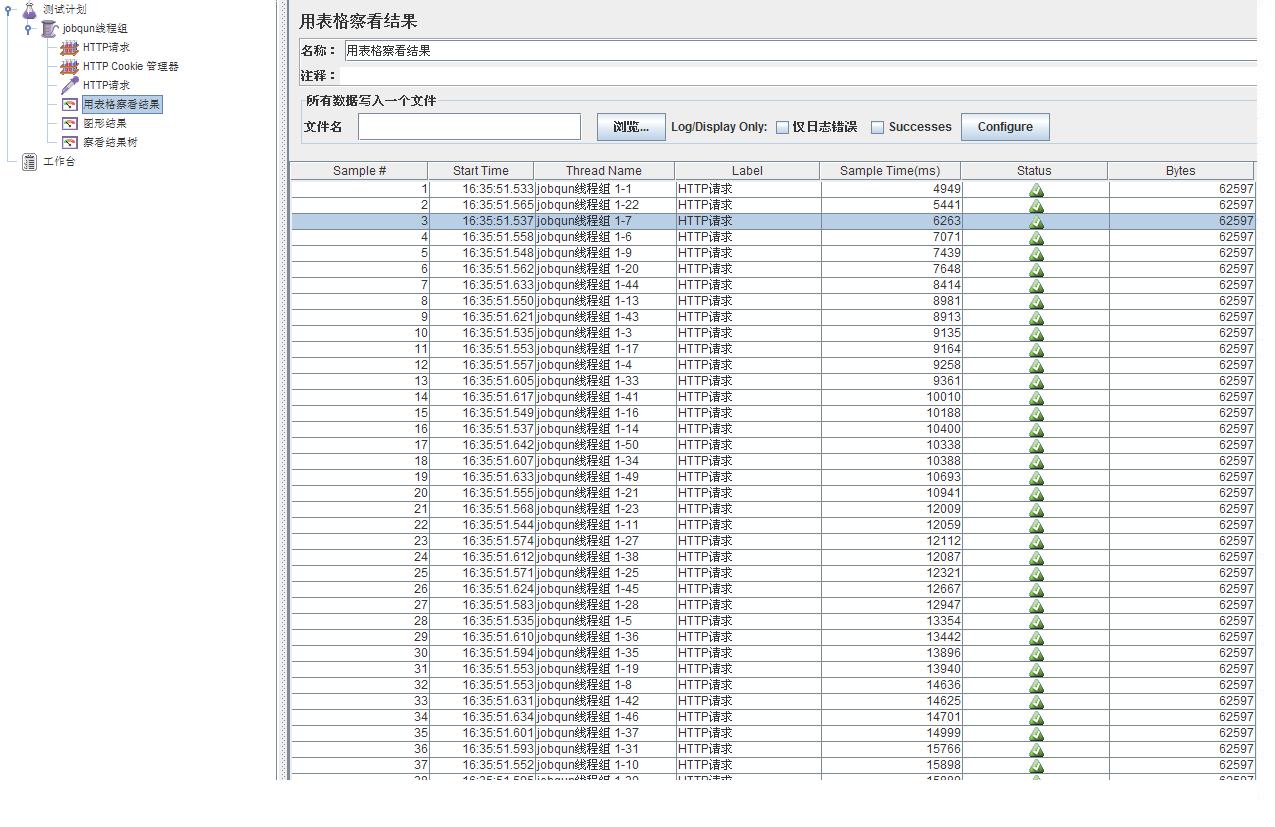
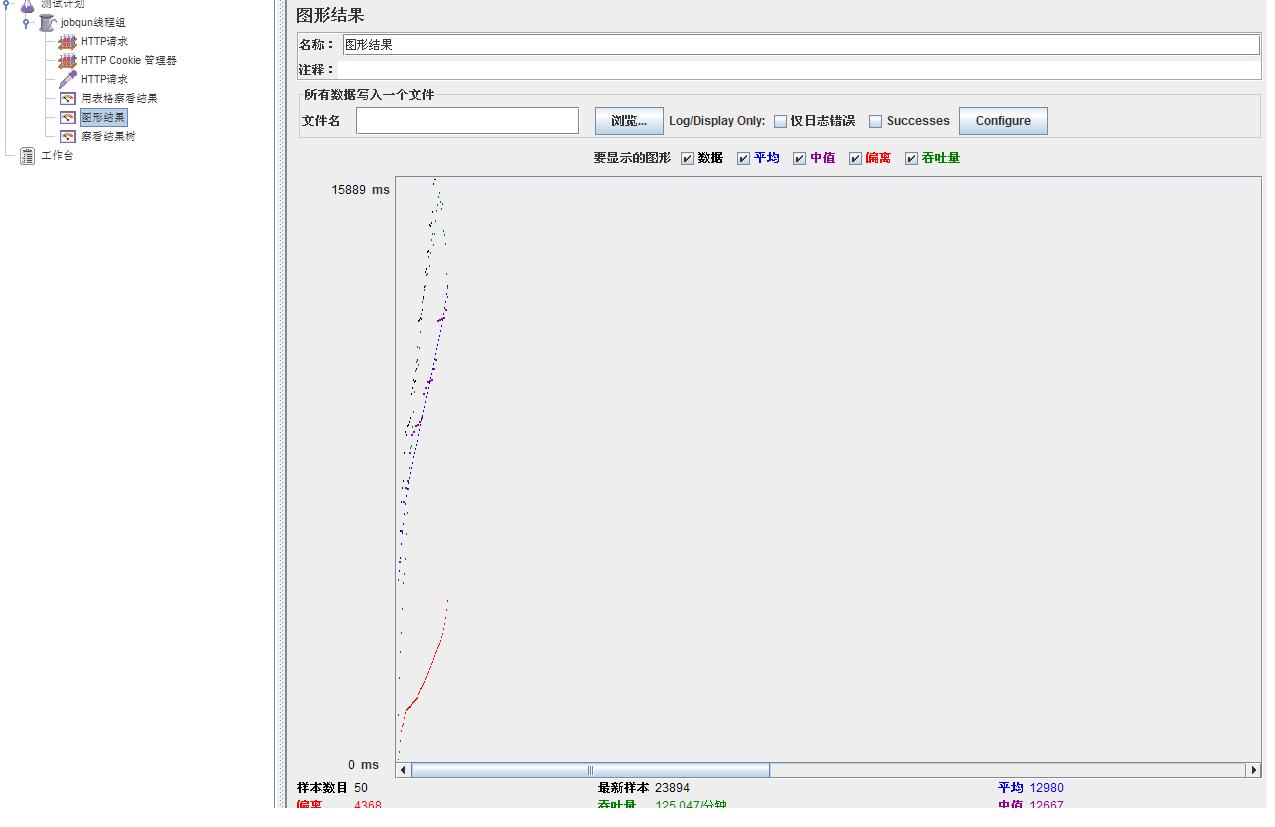
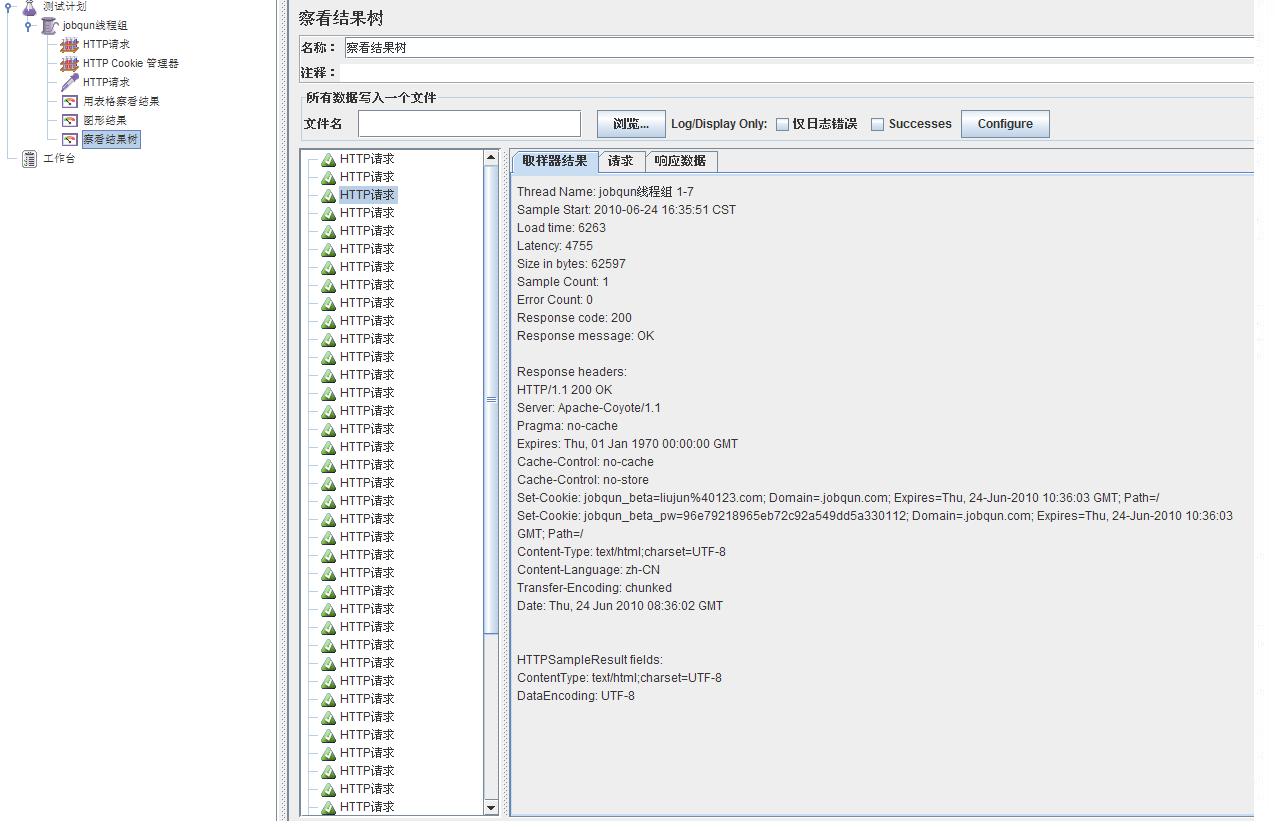
-.Set the Path field to "/changes.html".

(5). Add listener to view store test result: Select the JMeter Users 🡪Add --> Listener --> Graph Results

(6). Select Lisenter, Ctrl+R

1. Sprint3 [**jmeter 压力测试**](http://skly-java.iteye.com/blog/1682031)

--右击菜单添加线程组，如图1，

--并配置好参数如图2  
  
--然后在线程组上右击 添加Http请求默认值，它在这里类似于一个全局的，以后再配置类似HttpRequest的某些东西就不要再配置了，   
--服务器名称或ip配置为服务器的ip或者域名都可以  如果是tomcat的话  端口号为测试tomcat的端口号 content encoding为请求的编码格式  
--然后右击线程组添加http请求并配置好参数 如图4  
  
  这里的服务器名称和ip就不用再配置了  因为前面已经配置好全局的了。  然后把需要的参数添加进去，这里我测试的是登陆，所以添加了2个参数 一个loginEmail = [liujun@123.com](mailto:liujun@123.com) password = 111111。  
--然后右击线程组 依次添加用图表格查看结果、图形结果、查看结果树。  
最后ctr+R ，我们查看表格查看结果界面如图5  
  
 ，  同理 图6  
  
   图7  


下面是一些测试中的一些术语解释  
1.1       Aggregate Report  
Aggregate Report是 JMeter 常用的一个 Listener，中文被翻译为“聚合报告”。例如只有一个登录的请求，那么在Aggregate Report中，会显示一行数据，共有10个字段，含义分别如下。  
Label：每个 JMeter 的 element（例如 HTTP Request）都有一个 Name 属性，这里显示的就是 Name 属性的值  
#Samples：表示你这次测试中一共发出了多少个请求，如果模拟10个用户，每个用户迭代10次，那么这里显示100  
Average：平均响应时间——默认情况下是单个 Request 的平均响应时间

Median：中位数，也就是 50％ 用户的响应时间(ms)  
90% Line：90％用户的响应时间（ms）  
Min：最小响应时间  
Max：最大响应时间  
Error%：本次测试中出现错误的请求的数量/请求的总数  
Throughput：吞吐量——默认情况下表示每秒完成的请求数（Request per Second）  
KB/Sec：每秒从服务器端接收到的数据量  
1.2       用表格察看结果  
样本数目：总共发送到服务器的请求数  
最新样本：代表时间的数字,是服务器响应最后一个请求的时间。  
平均值：总运行时间除以发送到服务器的请求数。  
偏离：服务器响应时间变化、离散程度测量值的大小，或者，换句话说，就是数据的分布。  
1.3       图形报表  
样本数目：总共发送到服务器的请求数。  
最新样本：代表时间的数字,是服务器响应最后一个请求的时间。  
吞吐量：服务器每分钟处理的请求数。  
平均值：总运行时间除以发送到服务器的请求数。  
中间值：时间的数字，有一半的服务器响应时间低于该值而另一半高于该值。  
偏离：服务器响应时间变化、离散程度测量值的大小，或者，换句话说，就是数据的分布。

1. Sprint1- [Jmeter：图形界面压力测试工具](http://skly-java.iteye.com/blog/1681985)

1、  建立一个线程组，如下图



为什么要建立线程组？原因很简单，因为我们要模拟多个线程（用户）来访问网站。

线程组创建界面如下图：

名称可以随意填写，默认选中“继续”即可。

线程属性部分中，线程数是启动多少个线程，我这里填写的是60，Ramp-Up Period (in seconds)表示线程之间间隔多少时间允许，单位是秒，比如如果填写120，那么120/60=2表示60个线程间每隔2秒钟请求网站。

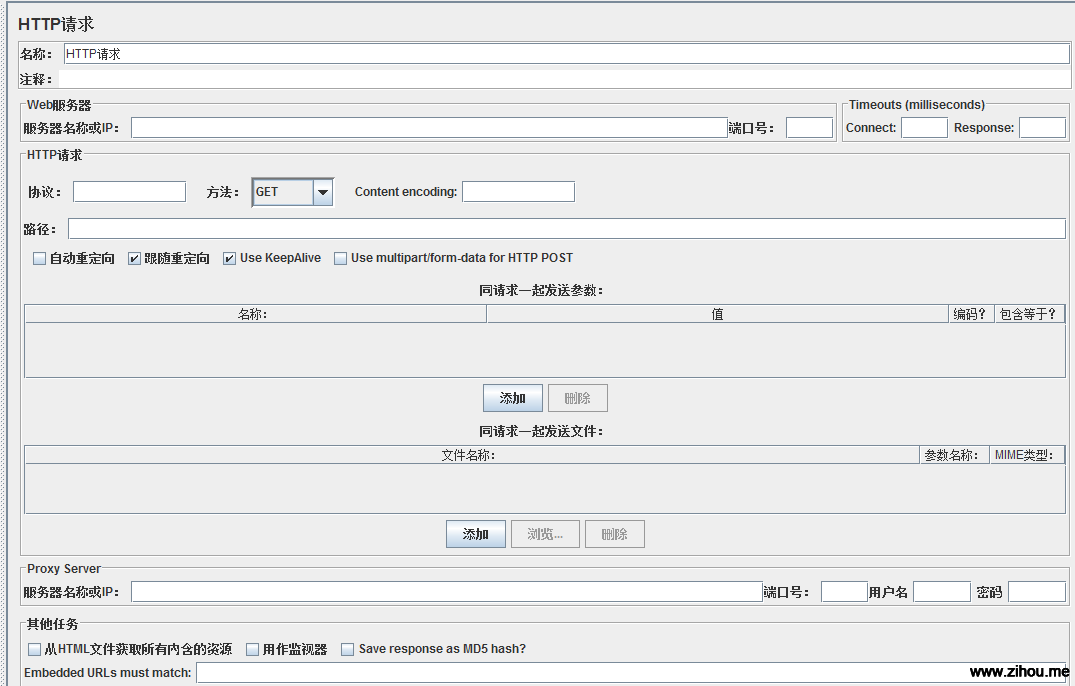
循环次数：60个线程运行完毕算是一次，循环次数就是这样的一个请求过程运行多少次，我这里填写的是1000.

每次修改一个设置后，别忘记了保存一下。

2、  设置请求服务器、压力链接等信息

接下来很自然的是，我们要测试的网站地址是什么？链接是什么？所以现在我们就来设置这些信息。

右键点击我们刚创建的线程组，在弹出的菜单中，选择添加->Sampler->Http请求，弹出如下图界面：



名称：随意填写

注释：可有可无

服务器名称或IP：我这里填写 zihou.me

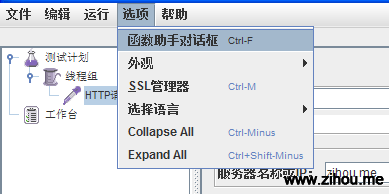
端口号：我这里填写80

Timesout部分可以不填

HTTP请求部分的协议：http，方法我选择的是GET，Content encoding我填的是UTF-8.

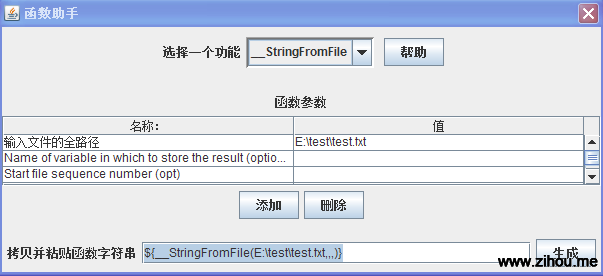
**路径**：

这里值得一提的就是这个路径，假如你只是对一个链接进行压力测试，那么这里就非常简单了，你就直接填写一个链接就够了，比如http://www.zihou.me，但很多情况下都不是这样的，我们这里需要多个链接，就如同刚开始讲到的那样，我们要将多个链接保存到一个文本文件中，然后随机读取进行压力测试。我们可以这么干，如图：

打开界面如下图：



在选择一个功能下拉列表中选择\_StringFromFile，然后在本机新建一个测试文件text.txt，在第一行（你也可以不在第一行）的值中填写测试文件的路径，如下图：



然后点击“生成”按钮，在生成按钮的左边文本框中将生成一个字符串如：

${\_\_StringFromFile(E:\test\test.txt,,,)}

在测试文件中，我们每行写一个URL链接，如下格式：

2011/03/26/3054

2011/03/26/3052

2011/03/26/3042

2011/03/25/3040

2011/03/25/3034

2011/03/24/3027

注意，每行前面并没有http://www.zihou.me这样的信息，因为我们在前面已经填写了服务器地址为zihou.me，这里就没必要再为每个url填写这个相同前缀了；另外，上面的url格式也只是个例子，表示域名后的部分，但很多时候url后面跟的都是各参数，比如http://www.zihou.me/p?a=1&b=3，如果是这种情况，则上面的链接可以写为：

a=1&b=3这样的形式。

现在我们可以来填写前面所说的路径了，如下：

/ ${\_\_StringFromFile(E:\test\test.txt,,,)}

这样一来，当我们并非请求的时候，就会从test.txt中随机选择url来进行压力测试。

另外值得注意的一个地方是，如果参数中有中文的情况，运行的时候可能会出现乱码，这个时候就需要注意你在Jmeter中的编码设置与你要请求的网页编码是一致的。

路径文本框下面的选项，可以按默认的就成，Use multipart/form-data for HTTP POST是当请求中有附件的情况，一般情况下都不用选中的。

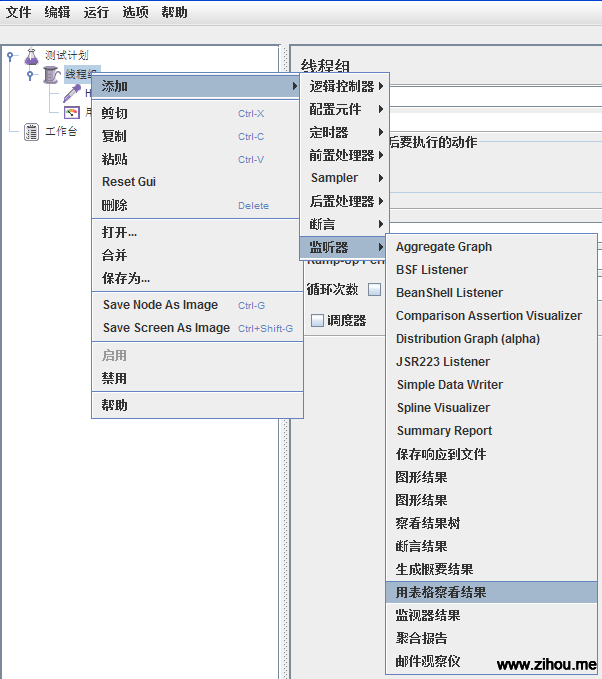
**同请求一起发送参数部分**：

如果你仅仅是对某一个固定的页面进行压力测试，那你就可以不用进行上面所说的在文本文件中设置URL的过程了，直接在这里设置就行了，这里的名称就是参数名，值就是参数值，在这里添加就OK了，但你如果是压力多个链接，那这里你可以不用填。

3、查看运行结果

上面设置好后，接下来很自然的想到如何查看运行结果呢？Jmeter在这方面提供了好些个查看方式，有表格形式，有曲线形式等等，我个人认为查看表格形式就足够了！

鼠标右键点击线程组，在弹出的菜单中选择添加->监听器->用表格查询结果，如下图：



弹出界面如下图：

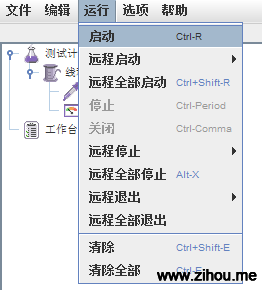


当然，你也可以选择其他查看方式，你也可以看到在监听器菜单中，有好多种方式。

**四、 运行**

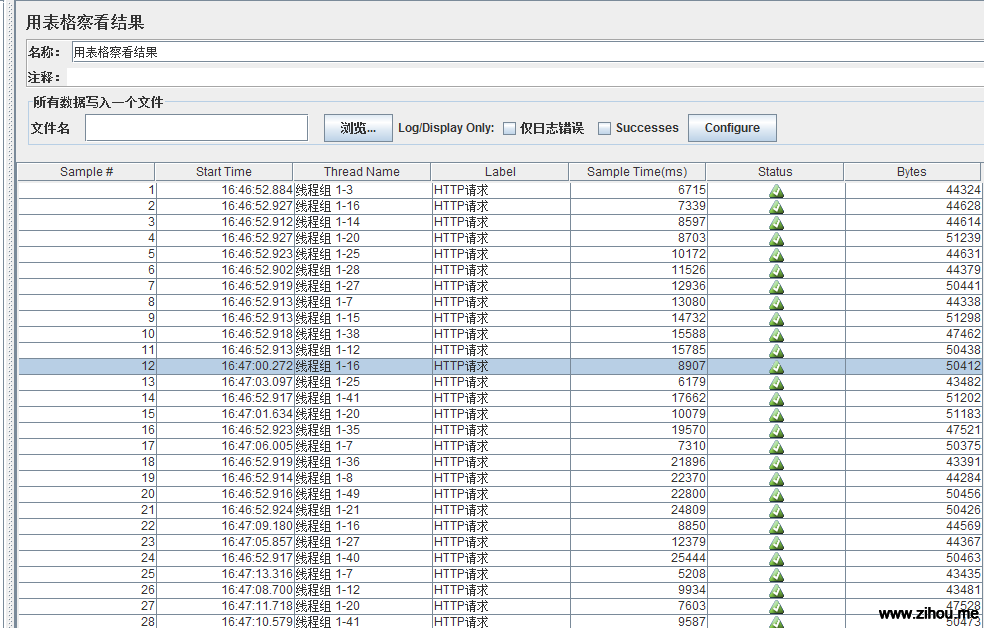
好了，现在我们终于可以运行了！

选择菜单项运行->启动，如下图：



当然，在运行前，你得把所有修改都保存了。

运行后的结果表格如下：



各属性如下：

Sample：每个请求的序号

Start Time：每个请求开始时间

Thread Name：每个线程的名称

Label：Http请求名称

Sample Time：每个请求所花时间，单位毫秒

Status：请求状态，如果为勾则表示成功，如果为叉表示失败。

Bytes：请求的字节数

如果Status为叉，那很显然请求是失败了，但如果是勾，也并不能认为请求就一定完全成功了，因为还得看Bytes的字节数是否是所请求网页的正常大小值，如果不是则说明发生了丢包现象，也不是完全成功。

在下面还有几个参数

样本数目：也就是上面所说的请求个数，成功的情况下等于你设定的并发数目乘以循环次数。

平均：每个线程请求的平均时间

最新样本：表示服务器响应最后一个请求的时间

偏离：服务器响应时间变化、离散程度测量值的大小，或者，换句话说，就是数据的分布（这个我不是很理解）。

在上面的参数中，我个人认为只要看Status和Bytes这两个就够了。

最后还提到的一点是：你一定注意到了，HTTP请求和结果查看都是在线程组这一节点下创建的，其实，你也可以不用一定要在线程组下创建，你在测试计划下建立也可以，但创建在线程组下面可以使得我们的测试意图一目了然，也就是我们的HTTP请求和结果查看都是基于我们设定的线程组的。

1. Sprint5-[**使用JMeter创建数据库（Mysql）测试**](http://skly-java.iteye.com/blog/1682172)

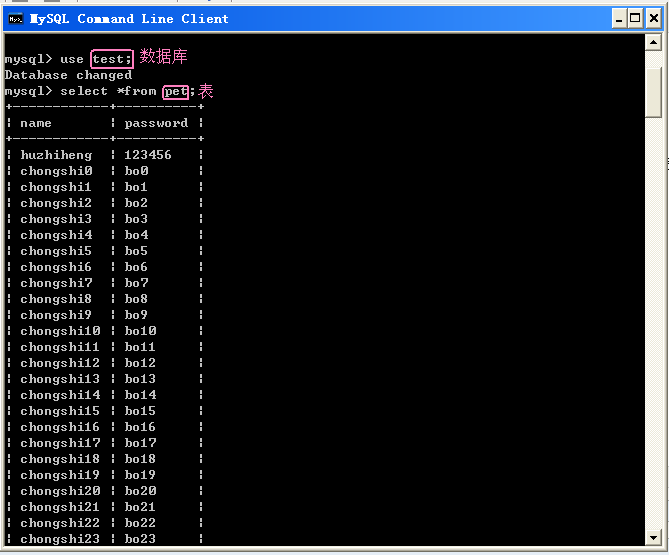
原文：<http://www.cnblogs.com/fnng/archive/2011/07/26/2117600.html>

我的环境：MySQL：sugarMysql

　　　　　jdbc驱动：bin\jdbc-connector-mysql

1.首先我们要有一个可以做测试的数据库，当然，里面要有数据，不然怎么测呢？我的上一篇文章《**[Eclipse连接MySQL数据库（傻瓜篇）](http://www.cnblogs.com/fnng/archive/2011/07/18/2110023.html)**》，里面教你如何通过程序批量插入数据。

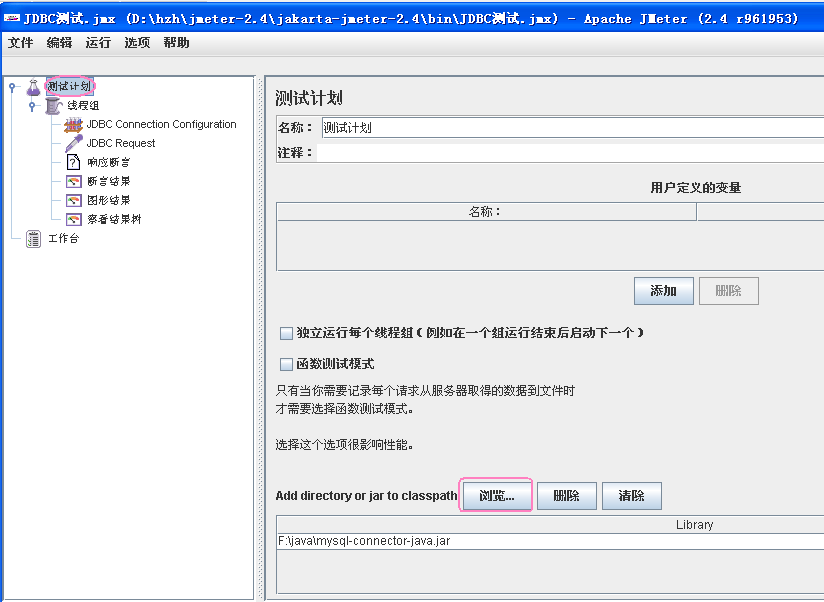
来看一下我的数据：



我可能有点啰嗦 :）  不管你用什么方式，你的数据库里要有点数据。

2.

打开JMeter,点击测试计划，



点击“浏览...”按钮，将你的JDBC驱动添加进来。

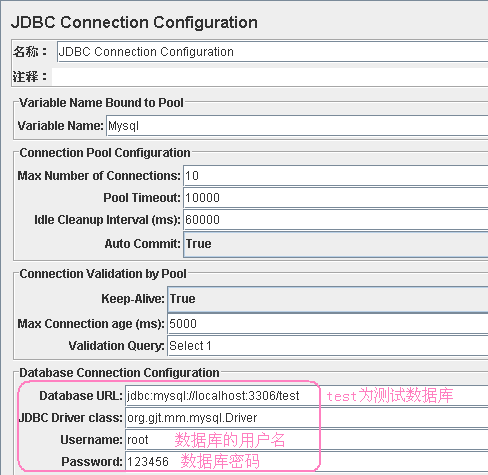
3.

添加一个线程组，

右键点击“线程组”，在下面添加一个“JDBC Connection Configuration”

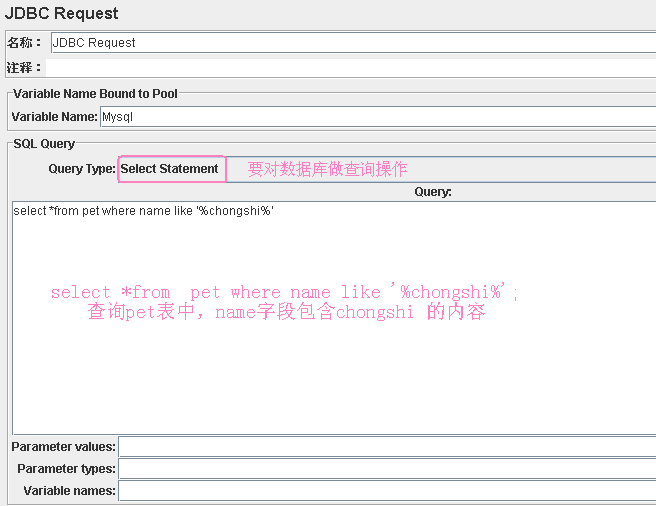


来配置一下JDBC Connection Configuration页面。



4.

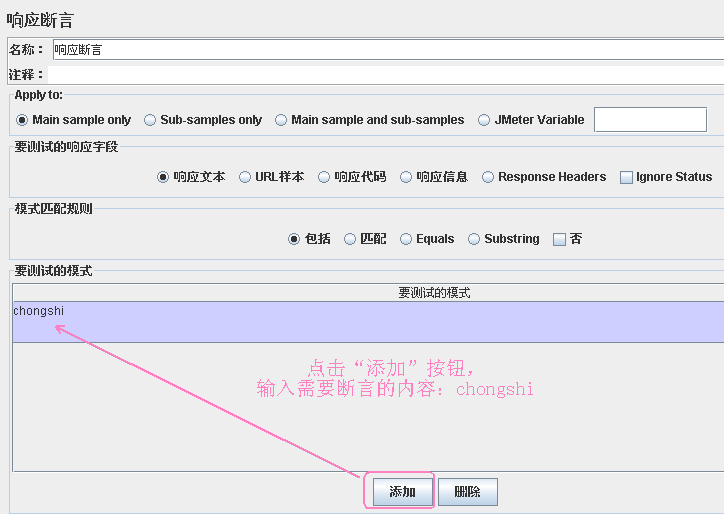
右键点击“线程组”，在下面添加一个“JDBC request”



5.

添加断言。

右键点击线程组---->添加--->断言---->响应断言。



6.

我们来添加一些监听器来行查看

添加一个断言结果：

右键点击线程组---->添加--->监听器---->结果断言。

添加一个图形结果：

右键点击线程组---->添加--->监听器---->图形结果。

添加一个查看结果树：

右键点击线程组---->添加--->监听器---->查看结果树。

下面是添加所有东东的列表：

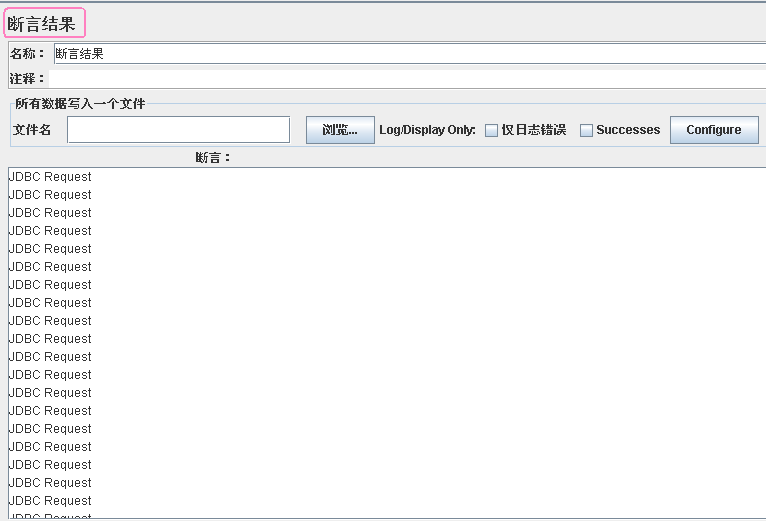


7.

在线程组页面设置用户数、启动时间、循环次数

点击菜单栏“运行”----“启动”

下面是结果：



1. Sprint6 -http接口测试——Jmeter接口测试实例讲解

**一、测试需求描述**

   1、 本次测试的接口为http服务端接口

   2、 接口的主要分成两类，一类提供给查询功能接口，一类提供保存数据功能接口，这里我们举例2个保存数据的接口，因为这两个接口有关联性，比较有代表性；

    3、接口描述：

**保存信用卡账户信息接口**：

            传入参数：

                args={

                    "clientNo":"434343556",

                    "alias": "\*\*信用卡2",

                    "cardName": "长城\*\*\*\*\*卡2",

                     "cardNo": "25622356788251",

                    }

            传出参数：

                    保存成功：{"returnCode":"0","returnMsg":"保存成功"}

                    保存失败：{"returnCode":"1","returnMsg":"保存失败"}

            保存逻辑：数据传入进来，验证通过，保存到信用卡账户表中

**保存信用卡账单接口：**

            传入参数：

                args={

                    "clientNo":"434343556",

                    "accountName": "测试",

                    "billDate": "08",

                    "billMonth": "201509",

                     "cardNo": "25622356788251",

                    "currentPayment": "欠款459.80",

                    "paymentDate": "2015-09-25 09:00:00",

                    }

            传出参数：

                    保存成功：{"returnCode":"0","returnMsg":"保存成功"}

                    保存失败：{"returnCode":"1","returnMsg":"保存失败"}

                     保存逻辑：保存时先去信用卡信息表查看clientNo对应的表是否存在，如存在则数据校验通过，

                                       将数据保存进入信用卡账单表

**注**：这个保存逻辑在接口开发设计文档中可能没有写或写的不详细，这时要与开发接口人员或产品人员多多沟通去熟悉接口逻辑

二、测试分析

    1、**逻辑分析**

        1)、从保存逻辑上来看，这两个接口明显是有依赖关系的，所以我们先测试信用卡账户信息接口，再测试保存信用卡账单接口

        2)、接口传入的数据，最终是保存到数据库中，所以当接口返回保存成功的时候，我们也要去对应的数据库表中核对相应的数据(这里可以用jmeter链接数据库进行操作代理手工)

        3)、当出现保存失败的情况时，我们需要查看系统的日志，所以我们也要准备好查看日志的权限和地址

**2、测试工具的准备**

        1)、单个接口测试，我们使用火狐的插件poster

        2)、多个接口测试，我们使用Jmeter进行测试

**三、使用工具测试**

    1、Poster工具的使用方法见

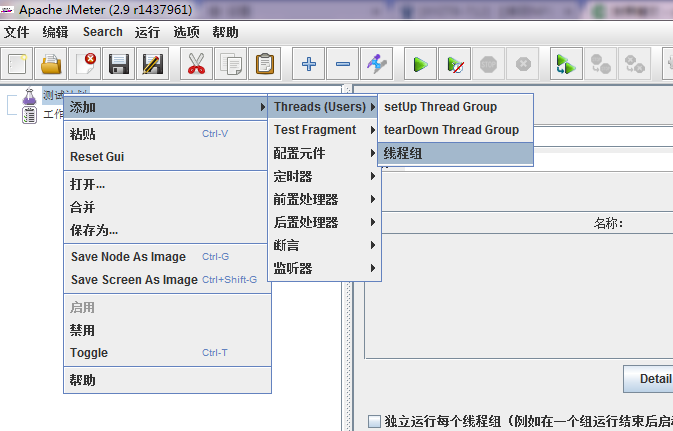
<http://note.youdao.com/share/?id=a3efa149d165258710ebf1bd04079f72&type=note>

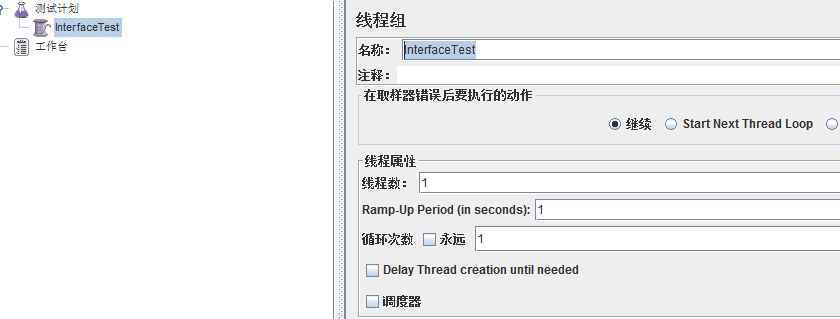
    2、使用Jmeter对接口测试

            首先我们说一下为什么用Poster测试后我们还要用Jmeter做接口测试，在用poster测试时候会发现的是一个接口一个接口的测试，我们每次测试 成功后的数据，在工具中是无法保存的，再次测试的时候我们还要重新输入测试的数据，当我们测试一个接口的时候可能感觉不明显，但是当你测试几十个接口的时 候，你就会发现使用Jmeter的好处，如果按测试阶段来说冒烟测试我们用poster，集成测试我们用Jmeter

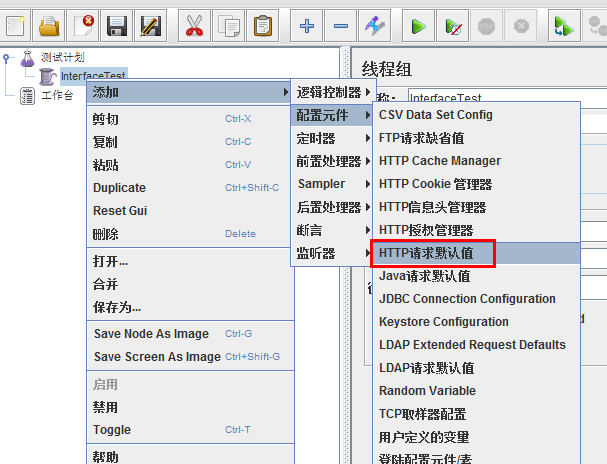
**四、使用Jmeter接口测试**

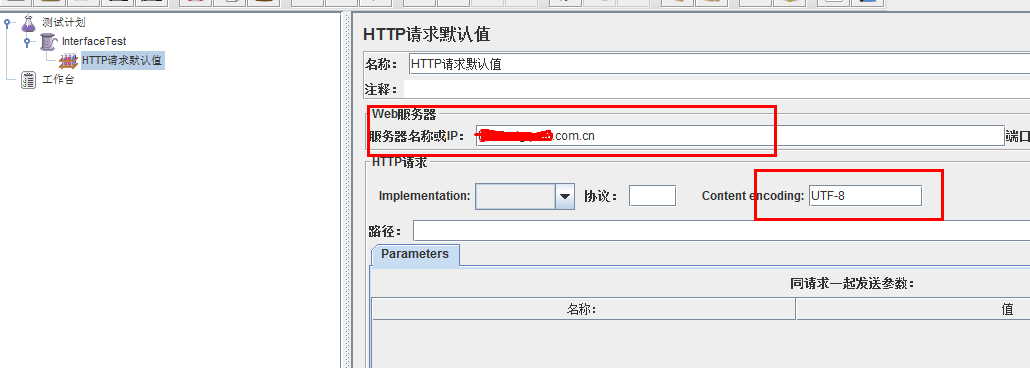
        1、首先邮件添加一个线程组，这里我们重命名InterfaceTest

[](http://static.oschina.net/uploads/space/2015/0929/185303_67iu_1391889.png)

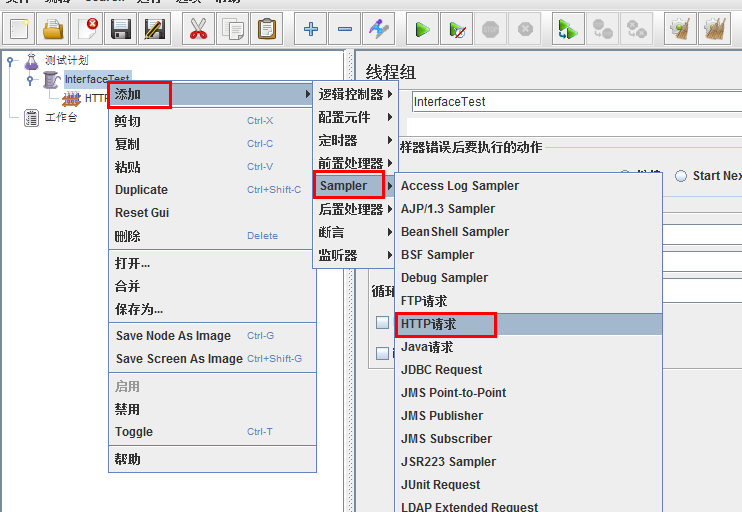
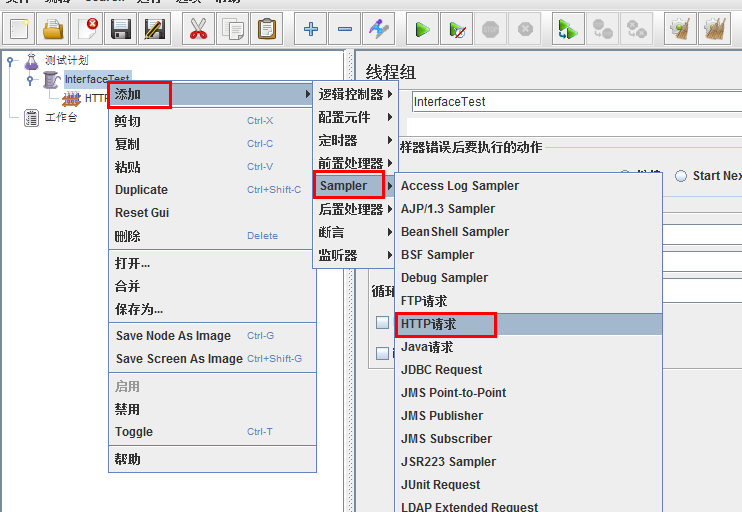
[](http://static.oschina.net/uploads/space/2015/0929/185339_6ids_1391889.png)

        2、在线程组上添加一个Http默认请求，并配置服务器的IP地址和传输编码

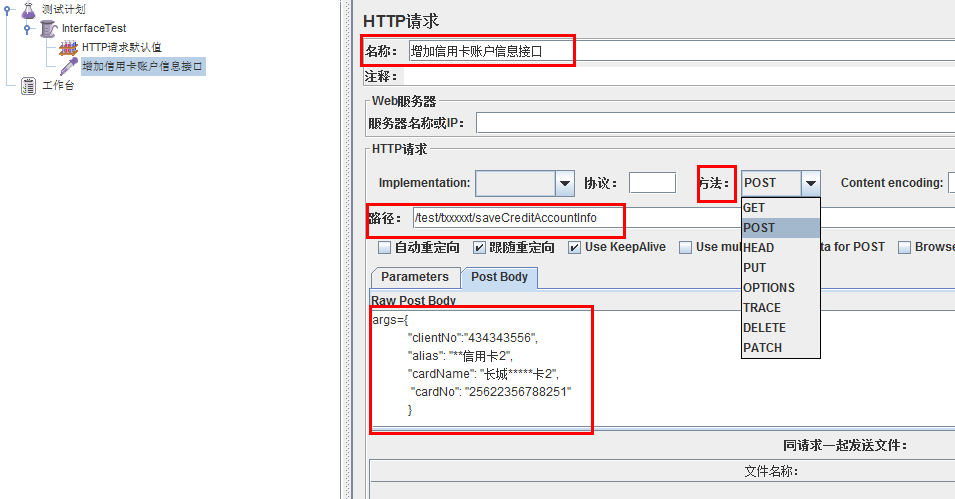
[](http://static.oschina.net/uploads/space/2015/0929/185441_r3Jk_1391889.png)

[](http://static.oschina.net/uploads/space/2015/0929/185441_a4M7_1391889.png)

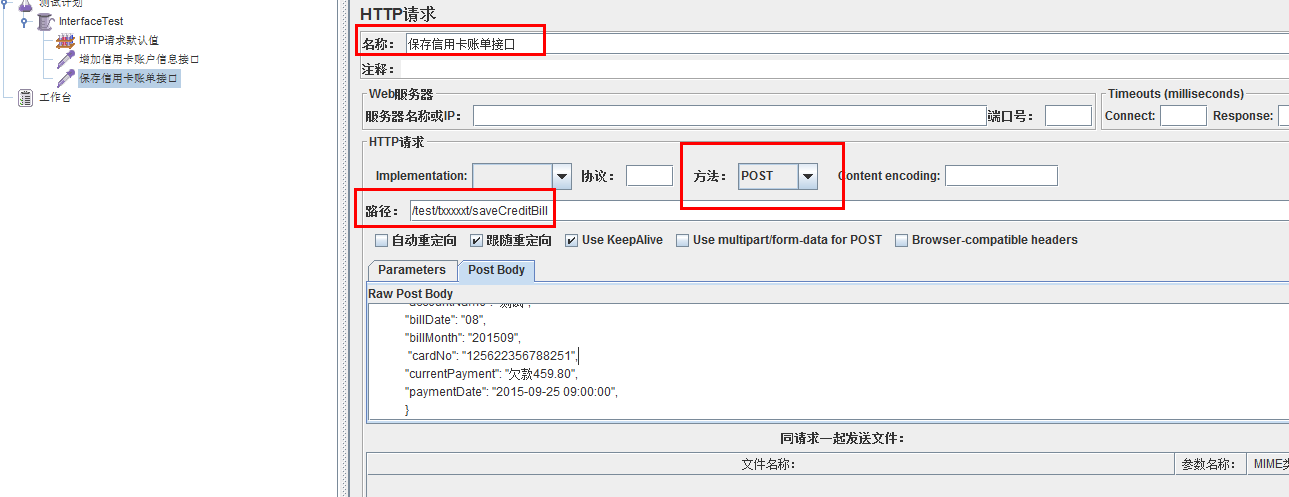
        3、在线程组中添加一个HTTP请求，这里我们重命名“增加信用卡账户信息接口”

[](http://static.oschina.net/uploads/space/2015/0929/185457_bVhj_1391889.png)    [](http://static.oschina.net/uploads/space/2015/0929/185600_kuJ2_1391889.png)

        4、配置接口请求信息，这配置示例如下：

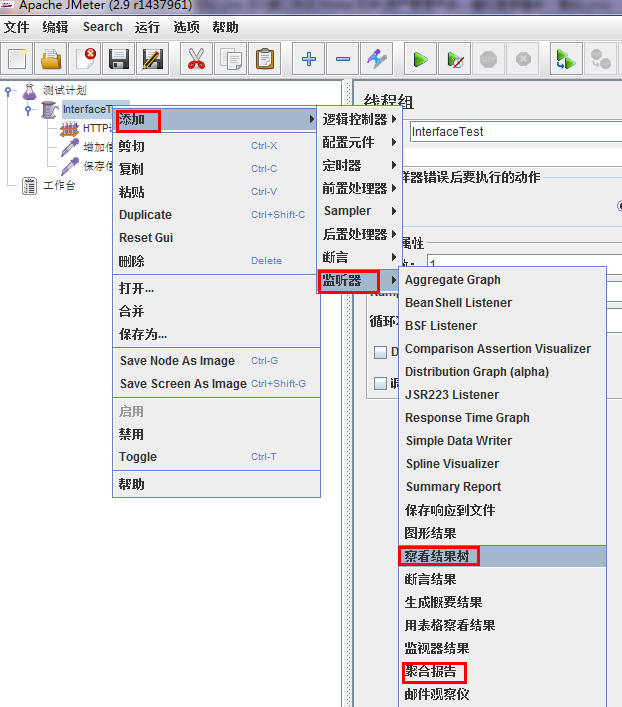
[](http://static.oschina.net/uploads/space/2015/0929/185631_lFg8_1391889.png)

        5，在保存信用卡账单接口请求，示例如下：

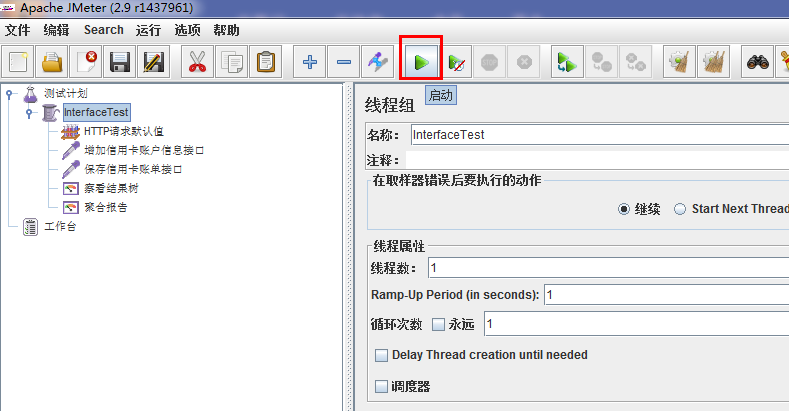
[](http://static.oschina.net/uploads/space/2015/0929/185709_HosX_1391889.png)

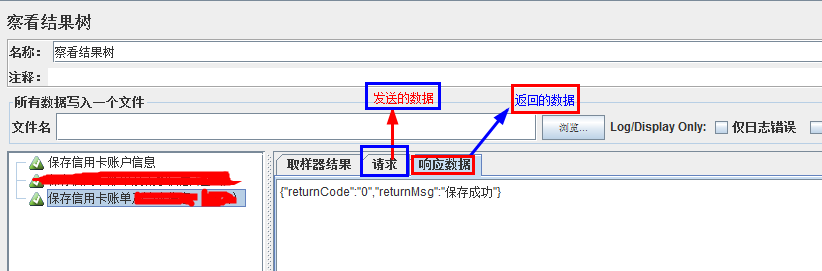
        注：由于Jmeter请求线程组内的请求时从第一个开始执行，所以我们将需要最先执行的请求放在前面

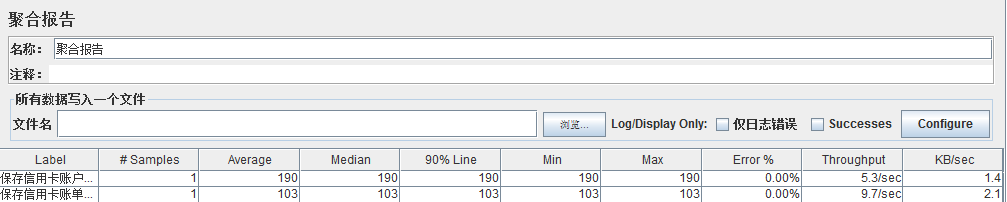
        6、在线程组上添加监听器，察看结果树和聚合报告

[](http://static.oschina.net/uploads/space/2015/0929/185737_GpUx_1391889.png)

        7、点击启动，运行结束后查看，结果树和聚合报告

[](http://static.oschina.net/uploads/space/2015/0929/185824_Vuru_1391889.png)

[](http://static.oschina.net/uploads/space/2015/0929/185908_BDCE_1391889.png)

[](http://static.oschina.net/uploads/space/2015/0929/185908_P1vT_1391889.png)

        8、去数据库中核对数据

        9、大批量数据制造

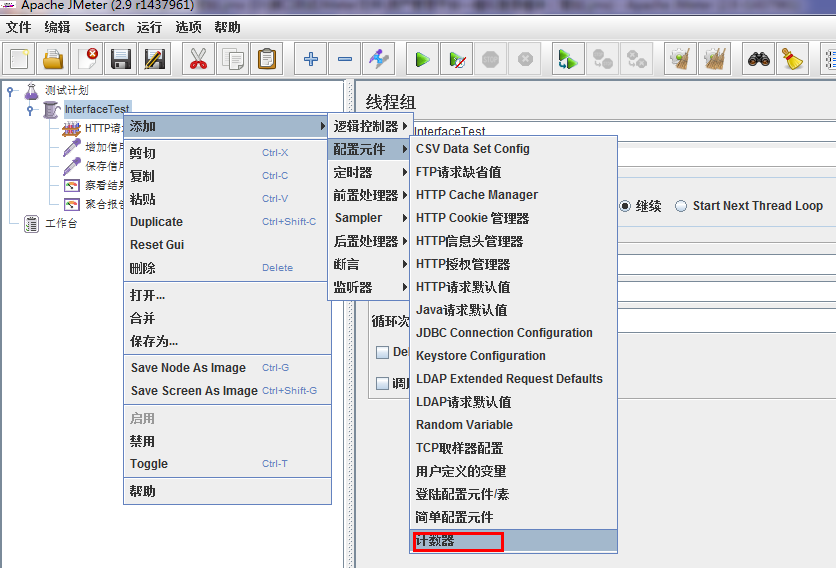
            思路：

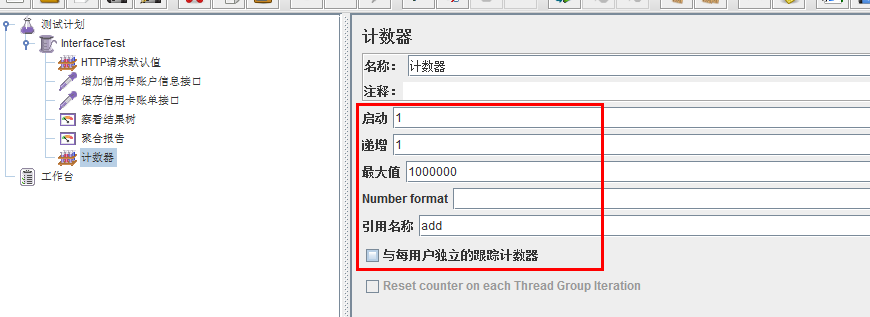
        1)、可参数化的参数，保存信用卡账户信息接口(clientNo,cardNo），保存信用卡账单接口(clientNo,cardNo,billMonth，paymentDate）

        2)、两个接口的依赖关系，保存信用卡账单接口(clientNo,cardNo）要和信用卡账户信息接口(clientNo,cardNo）的两个相同，也就是说这两个要用一个参数，且还不能重复

        根据上面两个特点，(clientNo,cardNo）我们选取使用计数器，每循环一次计数器加1，那么我们将线程组设置循环执行1万次；billMonth，paymentDate，这两个日期我们是使用随机函数${\_\_Random(1,9,)}，将月份参数化；

        3)、在线程组上创建计数器，配置如下：

[](http://static.oschina.net/uploads/space/2015/0929/185958_jLXw_1391889.png)

[](http://static.oschina.net/uploads/space/2015/0929/185958_y1ft_1391889.png)

        4)、引用计数器和随机函数

        信用卡账户接口传入参数

                args={

                    "clientNo":"${add}434343556",

                    "alias": "\*\*信用卡2",

                    "cardName": "长城\*\*\*\*\*卡2",

                     "cardNo": "${add}25622356788251",

                    }

        账单接口传入参数

        args={

                    "clientNo":"${add}434343556",

                    "accountName": "测试",

                    "billDate": "08",

                    "billMonth": "20150${\_\_Random(1,9,)}",

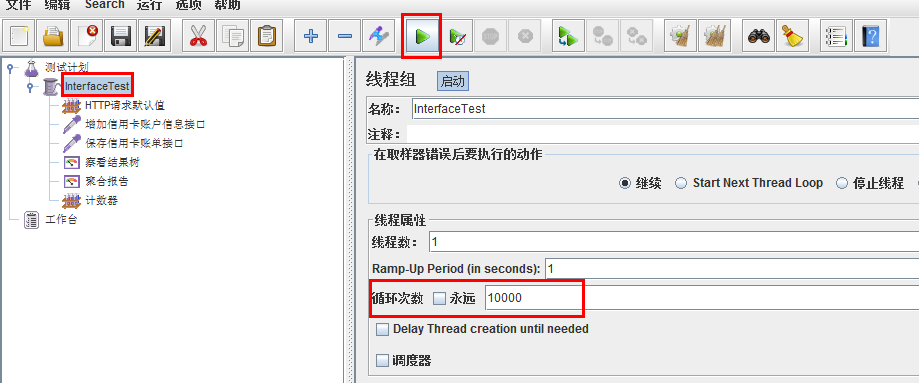
                     "cardNo": "${add}25622356788251",

                    "currentPayment": "欠款459.80",

                    "paymentDate": "2015-0${\_\_Random(1,9,)}-25 09:00:00",

                    }

        5)、设置线程组循环测试，点击运行,查看运行结果，再去数据库看看，大批量数据就这样产生了

[](http://static.oschina.net/uploads/space/2015/0929/190034_WoC7_1391889.png)

# My Activities:

* Download Jmeter-maven-plugin
* Start Jmeter-maven-plugin, 完成通过MAVEN启动Jmeter
* 我们所要做的事情就是如何解析Jmeter运行后所得到的测试结果，并将这个测试结果以相关的格式展现出来。

# Diagram

# All excerpt

这两天整理了一下文档，今天把JMeter录制脚本梳理成了word文档（有些地方都有截图，用word格式比较好记录），这里只把里面的文字部分拷出来了。

JMeter使用代理服务器录制脚本，对于需要使用VPN的录制不到脚本。

1、 运行jmeter.bat。

2、 右键单击“**[测试](javascript:;" \t "_self)**计划”，添加->Threads(Users)->线程组；右键单击添加好的线程组，添加->逻辑控制器->录制控制器。

3、 右键单击“**[工作](javascript:;" \t "_self)**台”，添加->非测试元件->HTTP代理服务器。参数设置：端口默认8080，目标控制器选“测试计划->线程组”（表示脚本生成在这个线程组下），分组选“每个组放入一个新的控制器”（表示把每组脚本放到一个新组中，避免生成的脚本杂乱无法组织），包含模式和排除模式用来过滤请求。

4、 点击最下方的“启动”按钮。

5、 打开浏览器，工具->Internet选项->连接->局域网设置->代理服务器。选中“为LAN使用代理服务器”，地址填本机地址127.0.0.1或localhost，端口号填8080（和步骤3中的HTTP代理服务器中的端口号一致）

6、 输入需要测试页面的URL，在打开的页面进行操作，JMeter即会将所进行的操作录制成脚本；可以看到在“线程组”下看到新加的一系列节点。

7、操作完成后，点击“停止”按钮停止录制，然后把浏览器的设置还原。

8、“录制控制器”这个元件可以先删除，保存测试计划（.jmx）

**3、运行及记录结果设置** (1)完成脚本的录制后还需要定义你所需的结果设置才算完成了Test Plan的设置，首先要设2置Thread Group（相当于虚拟用户组）的属性，单击Thread Group，在右边的窗口中设置Number of Thread（相当于虚拟用户数）为5，Ramp-Up Period(in seconds)为1，Loop Cout选项取消选中Forever，在右边的框中输入2。以上设置表示共有5个虚拟用户，每隔1秒运行一个，迭代2次。  
 (2)要查看运行的结 果还需要加入Listener，选中Thread Group单击鼠标右键选择Listener->View Results in Table，通过这个视图可以查看到每个链接的执行情况，包括执行时间和成功与否，再加入Aggregate Report，通过这个视图可以查看每个Thread Group组的最大、最小和平均响应时间。加入View Results Tree，通过这个视图可以查看每个链接的执行情况，包括发出的请求和服务器返回的详细内容。  
 (3)OK，所有的设置暂到此结束，接下载可以运行了。  
**4、运行及结果查看** (1)OK，运行看看吧，由于设置运行时间较短，所以先选择View Results in Table，然后按下Ctrl+r键开始运行脚本，就可以看到每个请求的运行结果了，还可以单击**[其他](javascript:;" \t "_self)**两个结果视图实时查看运行结果。  
 说 了这么多相信读者已经对这个工具有些感性的认识了吧，简单、便捷、跨平台、随手拈来即可使用，是它最大的有点（占内存是Java无法避免的，出现最先一 幕）。不过我的本意是想说说使用使用Jmeter进行查询内容参数化的，没想到就简单使用也花费了这么多篇幅，那只有下一节再叙了。总的感觉这个工具比较 “爽”。

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

From Peter

1. JMeter 2.7 (Open Source): Performance testing (80%), Function testing (20).
2. JMeter manual: <http://jmeter.apache.org/usermanual/index.html>
3. Click [User Manual](http://jmeter.apache.org/usermanual/index.html)🡪Click [5. Building a Web Test Plan](http://jmeter.apache.org/usermanual/build-web-test-plan.html)
4. cd apache-jmeter-2.7\apache-jmeter-2.7\bin
5. Double click JMeter.bat

(1). Add User (add the ThreadGroup): click Test Plan, mouse right click add button to get the Add menu, and then select Add --> ThreadGroup.

(2). Add HTTP Request Defaults: selecting the JMeter Users (Thread Group) element. Click your right mouse button to get the Add menu, and then select Add --> Config Element --> HTTP Request Defaults

Enter

IP: jmeter.apache.org

(3). add HTTP Request: JMeter Users element (Thread Group) (Add --> Sampler --> HTTP Request). Then, select the HTTP Request element in the tree and edit the following properties

Following properties

.Change the Name field to "Home Page".

.Set the Path field to "/".

(4), add another HTTP Request: (Same as c step) .

.Change the Name field to "Changes".

.Set the Path field to "/changes.html".

(5). Add listener to view store test result: Select the JMeter Users 🡪Add --> Listener --> Graph Results

(6). Select Lisenter, Ctrl+R



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

Build A JMeter Web Test Plan: (5 users) x (2 requests) x (repeat 2 times) = 20 HTTP requests

1. Add Users (on test plan): Add --> ThreadGroup

add a [Thread Group](http://jmeter.apache.org/usermanual/test_plan.html#thread_group)  (the number of users you want to simulate: 5, how often the users should send requests: delay 1, and the how many requests they should send 2.)

1. Adding Default HTTP request property (on group) : Add --> Config Element --> HTTP Request Defaults (server name: jmeter.apache.org)
2. add the [HTTP Cookie Manager](http://jmeter.apache.org/usermanual/component_reference.html#HTTP_Cookie_Manager) (on group): Add --> Config Element --> HTTP Cookie Manager
3. adding the first [HTTP Request](http://jmeter.apache.org/usermanual/component_reference.html#HTTP_Request) (Home Page): Add --> Sampler --> HTTP Request
4. the first [HTTP Request](http://jmeter.apache.org/usermanual/component_reference.html#HTTP_Request) (): Add --> Sampler --> HTTP Request
5. add a [Graph Results](http://jmeter.apache.org/usermanual/component_reference.html#Graph_Results) listener for storing all of the results of your HTTP requests : Add --> Listener --> Graph Results

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

[**JMeter基础之—录制脚本**](http://skly-java.iteye.com/blog/1681502)

Jmeter 是一个非常流行的性能测试工具，虽然与LoadRunner相比有很多不足，比如：它结果分析能力没有LoadRunner详细；很它的优点也有很多：

开源，他是一款开源的免费软件，使用它你不需要支付任何费用，

小巧，相比LR的庞大（最新LR11将近4GB），它非常小巧，不需要安装，但需要JDK环境，因为它是使用java开发的工具。

功能强大，jmeter设计之初只是一个简单的web性能测试工具，但经过不段的更新扩展，现在可以完成数据库、FTP、LDAP、WebService等方面的测试。因为它的开源性，当然你也可以根据自己的需求扩展它的功能。

我觉得它更像一个瑞士军刀，小巧，且功能齐全。初次认识Jmeter的时候，我觉得它不好，是因为相比LR来说，它没有脚本录制功能，也许不是没有，只是我不知道，因为文档上介绍的是这样，我要做一个web性能测试的话，就手动的一个个添加循环控制器、http信息管理头、http请求等等各种元件。如果测试的脚本较多时，这无疑是个体力活。

Badboy是一款不错web自动化测试工具，利用它来录制脚本，并且录制的脚本可以直接保存为JMeter文件来使用。我无疑给我们带来了很大我方便。

----------------------我的环境------------

Badboy  version 2.1.1

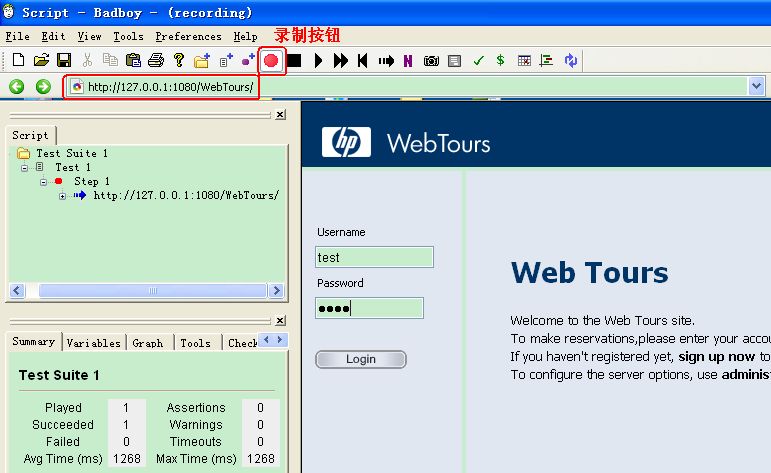
Apache  JMeter-2.3.4 (需要JDK环境来运行)

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

**第一种方法：通过bodboy来录制脚本。**

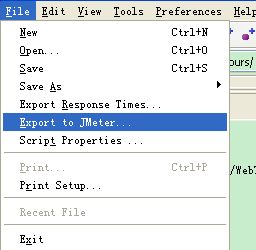
下载地址：<http://www.badboy.com.au/>

1.打开人badboy工具，点击工栏目上的红色圆形按钮，在地址栏目中输入被测试项目的地址。

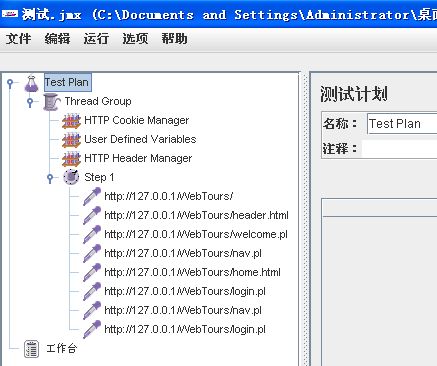


录制完成后，点击工具栏旁边黑色按钮，结束录制。

选择“文件”--Export to Jmeter…



2.打开Jmeter工具，选择“文件”--“打开”选择刚才保存的文件(.jmx类型)，将文件导入进来了。



**第二种方法，通过JMeter自身设置来录制脚本。**

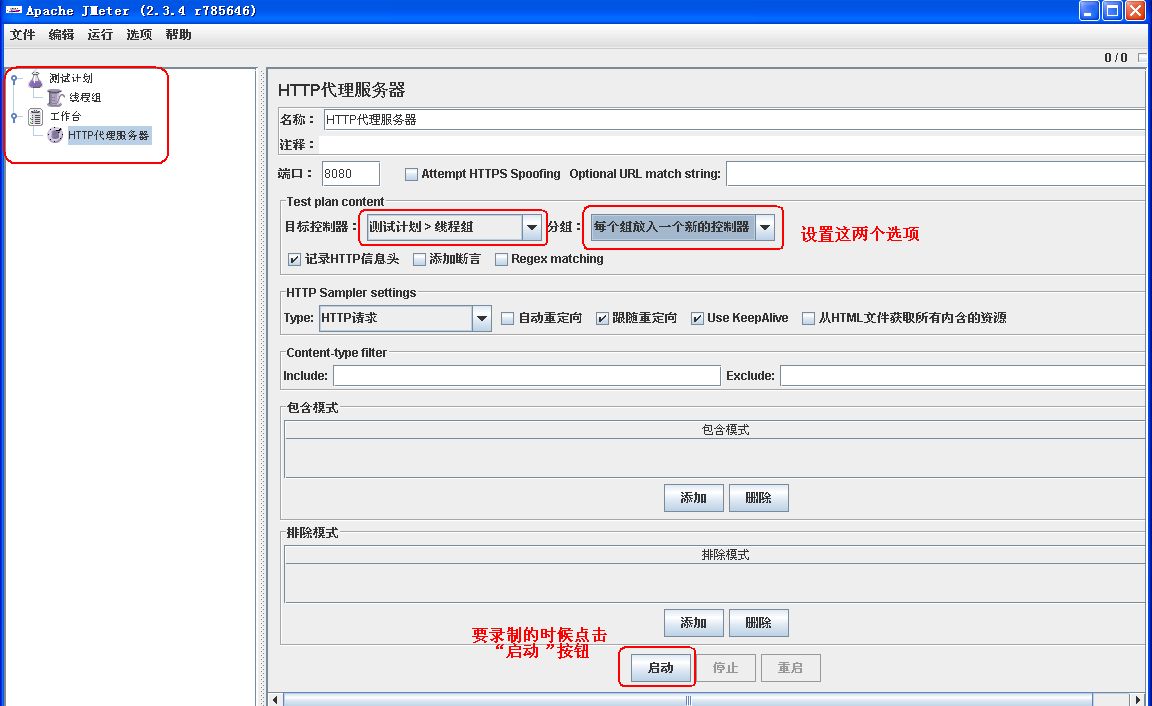
这种方法是我才发现的（鄙视一下自己的无知，嘻嘻~！），觉得方法比较简单。

<!--[if !supportLists]-->1.       <!--[endif]-->打开JMeter工具

创建一个线程组（右键点击“测试计划”--->“添加”---->“线程组”）

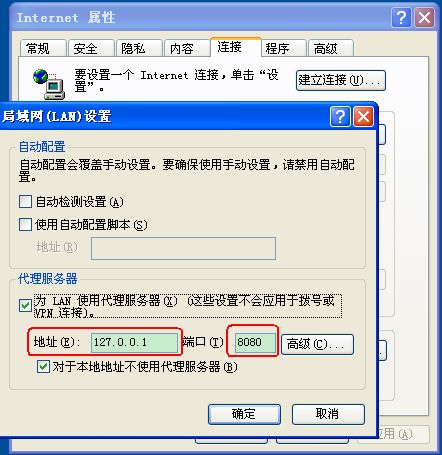
创建一个http代理服务器（右键点击“工作台”--->“添加”--->“非测试元件”--->“http代理服务器”）

完整的设置参照下图：



2.  下面来设置一下IE浏览器

IE--->“internet属性”--->“连接”--->“局域网设置”



设置为本机IP就可以了，注意端口号要与Jmeter上的端口号一致。默认都是8080端口。

3.   现在点击jmeter上的“启动”按钮，打开浏览器输入需要录制web项目地址，jmeter会自动记录你IE所访问的页面。



PS:第二种方法是我刚才知道的一种，关于这两种方法哪个更好，现在还不知道，但第二方法有通过IE浏览器辅助的，我想可能只要IE能打开的，它都能记录，但它录制的脚本看上去比较乱（感觉上）

还就是http代理服务器的设置，（比如：分组：每一个组放入一个新的服务器---只有这一个选项才能正常录制），有时间再仔细比较一下两种方法的不同之处。

**Jmeter基础之---jmeter基础概念**

发表于：2013-08-12来源：博客园作者：虫师点击数：69611 标签：[jmeter](http://www.ltesting.net/tags.php?/jmeter/)

JMeter 介绍： 一个非常优秀的开源的性能测试工具。 优点：你用着用着就会发现它的重多优点，当然不足点也会呈现出来。

[JMeter](http://www.ltesting.net/ceshi/open/kyxncsgj/jmeter/) 介绍： 一个非常优秀的[开源](http://www.ltesting.net/ceshi/open/)的[性能HYPERLINK "http://www.ltesting.net/"测试](http://www.ltesting.net/ceshi/ceshijishu/xncs/)工具。

　　优点：你用着用着就会发现它的重多优点，当然不足点也会呈现出来。

　　从性能工具的原理划分：

[Jmeter](http://www.ltesting.net/ceshi/open/kyxncsgj/jmeter/)工具和其他性能工具在原理上完全一致，工具包含4个部分：

　　(1)负载发生器：用于产生负载，通常以多线程或是多进程的方式模拟用户行为。

　　(2)用户运行器：通常是一个脚本运行引擎，用户运行器附加在线程或进程上，根据脚本要求模拟指定的用户行为。

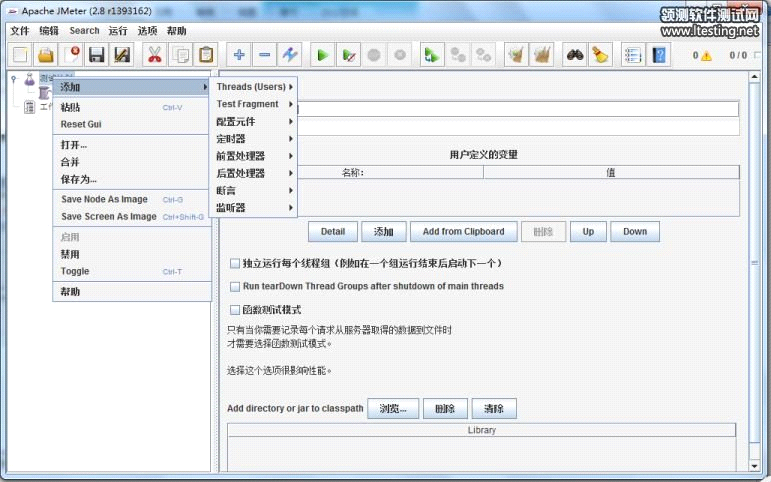
　　(3)资源生成器：用于生成测试过程中[服务器](http://www.ltesting.net/ceshi/ruanjianceshikaifajishu/rjcshj)、负载机的资源数据。

　　(4)报表生成器：根据测试中霍地的数据生成报表，提供可视化的数据显示方式。

　　测试计划元件

　　Test Plan (测试计划)：用来描述一个[性能测试](http://www.ltesting.net/ceshi/ceshijishu/xncs/)，包含与本次[性能测试](http://www.ltesting.net/ceshi/ceshijishu/xncs/)所有相关的功能。也就说本的性能测试的所有内容是于基于一个计划的。

　　下面看一下一个计划下面都有哪些主要的功能模块(右键单击“测试计划”弹出菜单)。



　　Threads (Users)线程 用户



　　虽然有三个添加线程组的选项，名字不一样， 创建之后，其界面是完全一样的。之前的版本只有一个线程组的名字。现在多一个setUp theread Group 与terDown Thread Group

　　1) setup thread group

　　一种特殊类型的ThreadGroup的，可用于执行预测试操作。这些线程的行为完全像一个正常的线程组元件。不同的是，这些类型的线程执行测试前进行定期线程组的执行。

　　2) teardown thread group.

　　一种特殊类型的ThreadGroup的，可用于执行测试后动作。这些线程的行为完全像一个正常的线程组元件。不同的是，这些类型的线程执行测试结束后执行定期的线程组。

　　可能你还是不太理他们与普通的线程组有什么不同。 如果您用过junit，想必你不会对setup ，teardown这2个字眼陌生。 即时每用过，也没关系。 熟悉loadrunner的应该知道，loadrunner的脚本除了action里是真正的脚本核心内容，还有初始化“环境”的初始化脚本和测试完毕后对应的清除信息的脚本块。 那么这里 setup thread group 和 teardown thread group 就是分别指这两部分。 其实从本质上来看，他们并没有什么不同。

　　3) thread group(线程组).

　　这个就是我们通常添加运行的线程。通俗的讲一个线程组,，可以看做一个虚拟用户组，线程组中的每个线程都可以理解为一个虚拟用户。线程组中包含的线程数量在测试执行过程中是不会发生改变的。

　　测试片段(Test Fragment)



　　测试片段是在2.5版本之后新加的一个选项。

　　测试片段元素是控制器上的一个种特殊的线程组，它在测试树上与线程组处于一个层级。它与线程组有所不同，因为它不被执行，除非它是一个模块控制器或者是被控制器所引用时才会被执行。

　　控制器

　　JMeter有两种类型的控制器：取样器(sample)和逻辑控制器(Logic Controller)，用这些原件来驱动处理一个测试。

　　取样器(Sampler)



　　取样器(Sample)是性能测试中向服务器发送请求，记录响应信息，记录响应时间的最小单元，JMeter 原生支持多种不同的sampler ，如 HTTP Request Sampler 、 FTP Request Sample 、TCP Request Sample 、JDBC Request Sampler 等，每一种不同类型的 sampler 可以根据设置的参数向服务器发出不同类型的请求。(在jmeter 的所有sampler 中，[Java](http://www.ltesting.net/ceshi/ruanjianceshikaifajishu/rjcskf) Request Sampler 和 Beanshell Request Sampler 是两种特殊的可定制的 Sampler ，后面会深入讨论。)

　　逻辑控制器(Logic Controller)



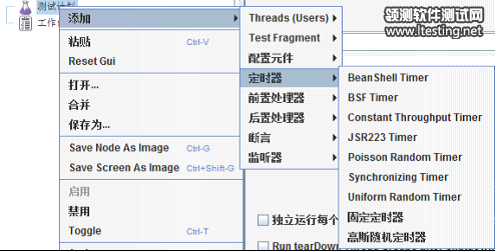
　　逻辑控制器，包括两类无件，一类是用于控制test plan 中 sampler 节点发送请求的逻辑顺序的控制器，常用的有 如果(If)控制器 、switch Controller 、Runtime Controller、循环控制器等。另一类是用来组织可控制 sampler 来节点的，如 事务控制器、吞吐量控制器。

　　配置元件(Config Element)



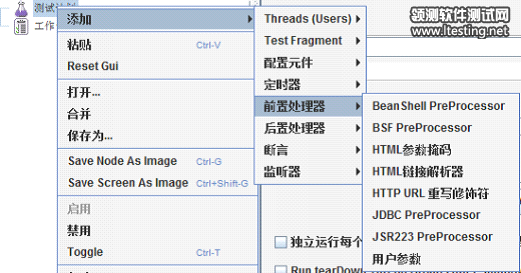
　　配置元件(config element)用于提供对静态数据配置的支持。CSV Data Set config 可以将本地数据文件形成数据池(Data Pool)，而对应于HTTP Request Sampler和 TCP Request Sampler等类型的配制无件则可以修改Sampler的默认数据。(例如，HTTP Cookie Manager 可以用于对 HTTP Request Sampler 的cookie 进行管理)

　　定时器(Timer)



　　定时器(Timer)用于操作之间设置等待时间，等待时间是性能测试中常用的控制客户端QPS的手端。类似于LoadRunner里面的“思考时间”。JMeter 定义了Bean Shell Timer、Constant Throughput Timer、固定定时器等不同类型的Timer。

　　前置处理器(Per Processors)



　　用于在实际的请求发出之前对即将发出的请求进行特殊处理。例如，HTTP URL重写修复符则可以实现URL重写，当RUL中有sessionID 一类的session信息时，可以通过该处理器填充发出请求的实际的sessionID 。

**如何使用Jmeter,maven,Jenkins构建云性能测试平台**

发表于：2013-02-16来源：未知作者：victorcai0922点击数：8769 标签：

如何使用Jmeter,maven,Jenkins构建云性能测试平台

[Jmeter](http://www.ltesting.net/ceshi/open/kyxncsgj/jmeter/)+maven+Jenkins构建云性能[测试](http://www.ltesting.net/)平台

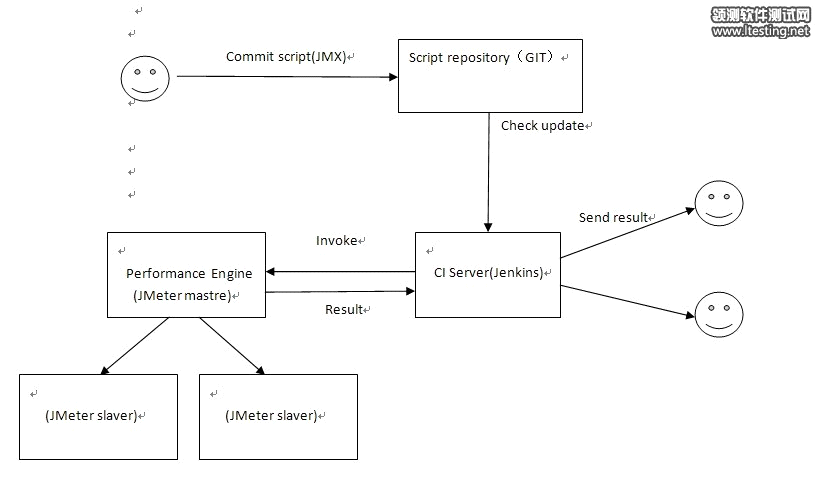
　　如何使用Jmeter,maven,Jenkins构建云性能测试平台,[软件测试HYPERLINK "http://www.ltesting.net/ceshi/ceshijishu/xncs/"性能测试](http://www.ltesting.net/)

　　最近在利用Jmeter来做一套自动化性能测试框架，做[自动化](http://www.ltesting.net/ceshi/ceshijishu/zdcs/)性能测试框架的目的是希望能够针对系统做一个benchmark的性能测试，能够快速的在每个版本发布后，对该版本进行benchmark性能测试，以比较与上一个版本的性能是否发生变化，若发生变化便可以快速的通知开发人员以确定性能发生变化的模块或者代码。同时测试或者[开发](http://www.ltesting.net/ceshi/ruanjianceshikaifajishu/)可以上传性能测试脚本到测试平台,测试平台接到测试脚本后就可以进行性能测试并返回测试结果.基于这一点的考虑是,一般的测试工程师(特别想功能测试[工程师](http://www.ltesting.net/ceshi/ceshijishu/rjcsgcsrm/))和开发工程师可能没有相关的性能测试环境供他们使用,因此建立这样的平台就是希望能够让每个人都能快速的进行性能测试,而不需要关注性能测试环境(往往配置一个性能环境可能需要很长的时候).

　　1.框架简要介绍

　　选择Jmeter来搭建这样一个平台是基于Jmeter本身的Open source, 并且是用纯JAVA开发的测试工具，方便与其他的工具的集成和扩展。最重要的是，你可以看到源码。

　　这套框架基本的执行流程和逻辑其实很简单，利用Jenkins的CI功能，实时检查脚本库，若有变更便调用JMeter来执行脚本，并将测试结果通过邮件或其他方式通知相关人员。主要逻辑图如下所示：

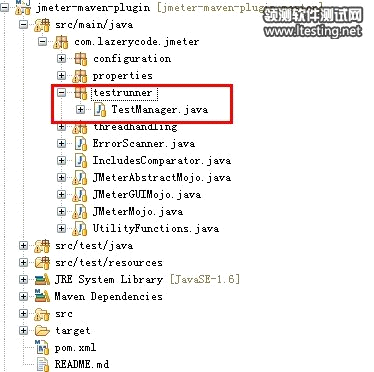


　　2. Maven执行Jmeter

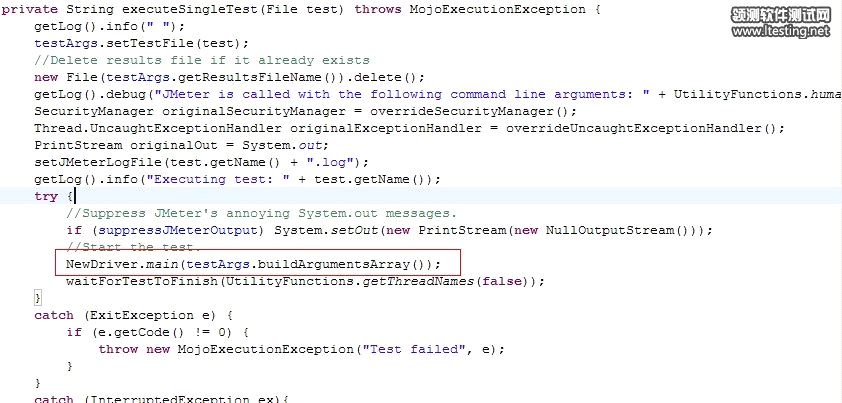
　　MAVEN是一个非常优秀的项目管理工具，关于Maven和Ant的主要区别可以去网上查询，我们在这里主要介绍一下用MAVEN如何去运行JMeter， JMeter支持多种运行方式，有GUI方式和NONGUI方式，各有优势，我们在自动化性能测试平台的搭建中采用NonGUI方式来运行测试脚本，NonGuI方式其实也就是通过Command命令来运行，那么如何通过Maven来调用呢，不用慌张，已经有Jmeter-maven-plugin这样一个Maven插件来运行Jmeter了，如果看过Jmeter源码的话，可以看到在Jmeter中有这样一个Class，叫做NewDriver.class，这个类是Jmeter的入口，我们可以看一下这个类的Main方法：



　　看到try模块中的最后几行可以看到，通过Java反射机制，JMeter.start()方法被调用到，并且将相关的参数传递给该方法。因此我们可以想象到JMeter-maven插件中肯定也是通过调用这个方法来启动JMeter的，我们来看一下JMeter-maven-plugin这个插件(关于如何开发maven插件在这里不具体讲，可以参考网上资料)中的主要调用代码：



　　重点参考TestManager这个类，这个类是主要用来启动Jmeter的，我们可以参考这个类中的executeSingleTest(File test)这个方法：



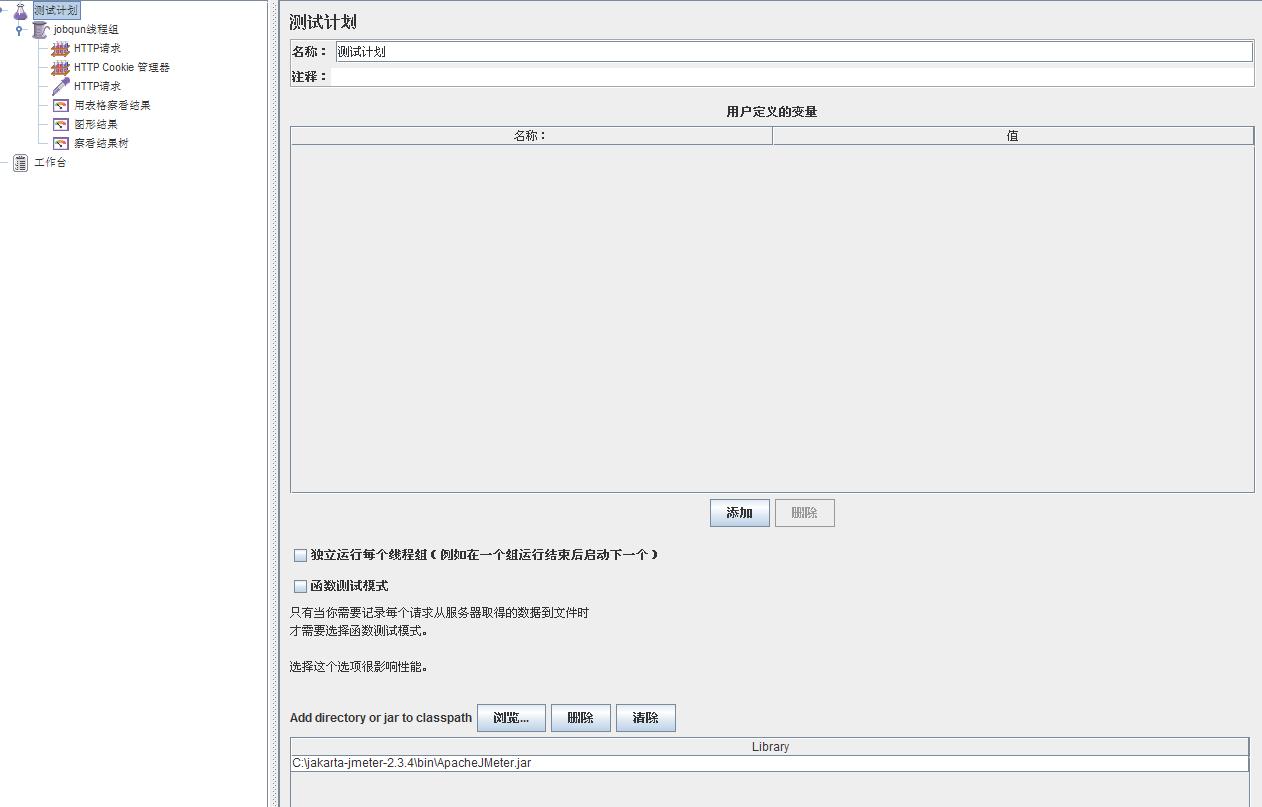
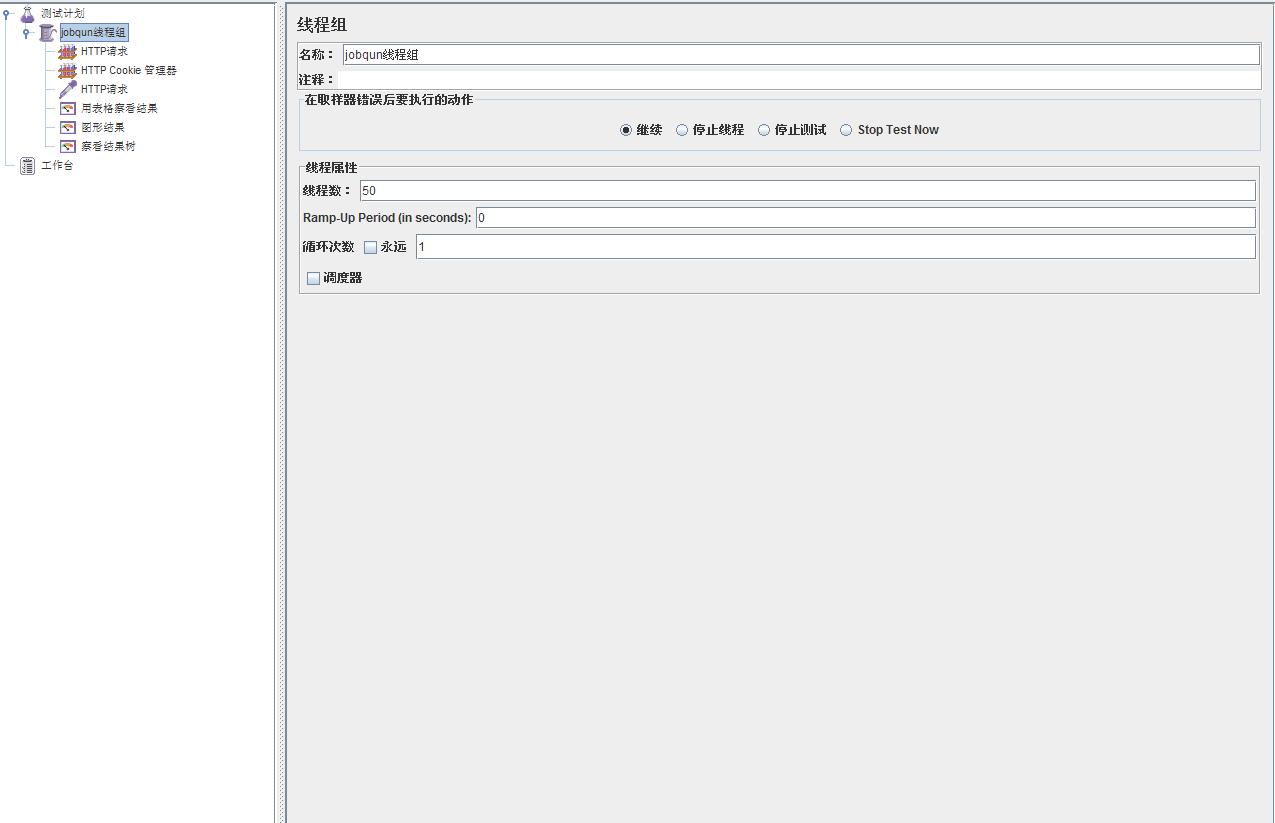
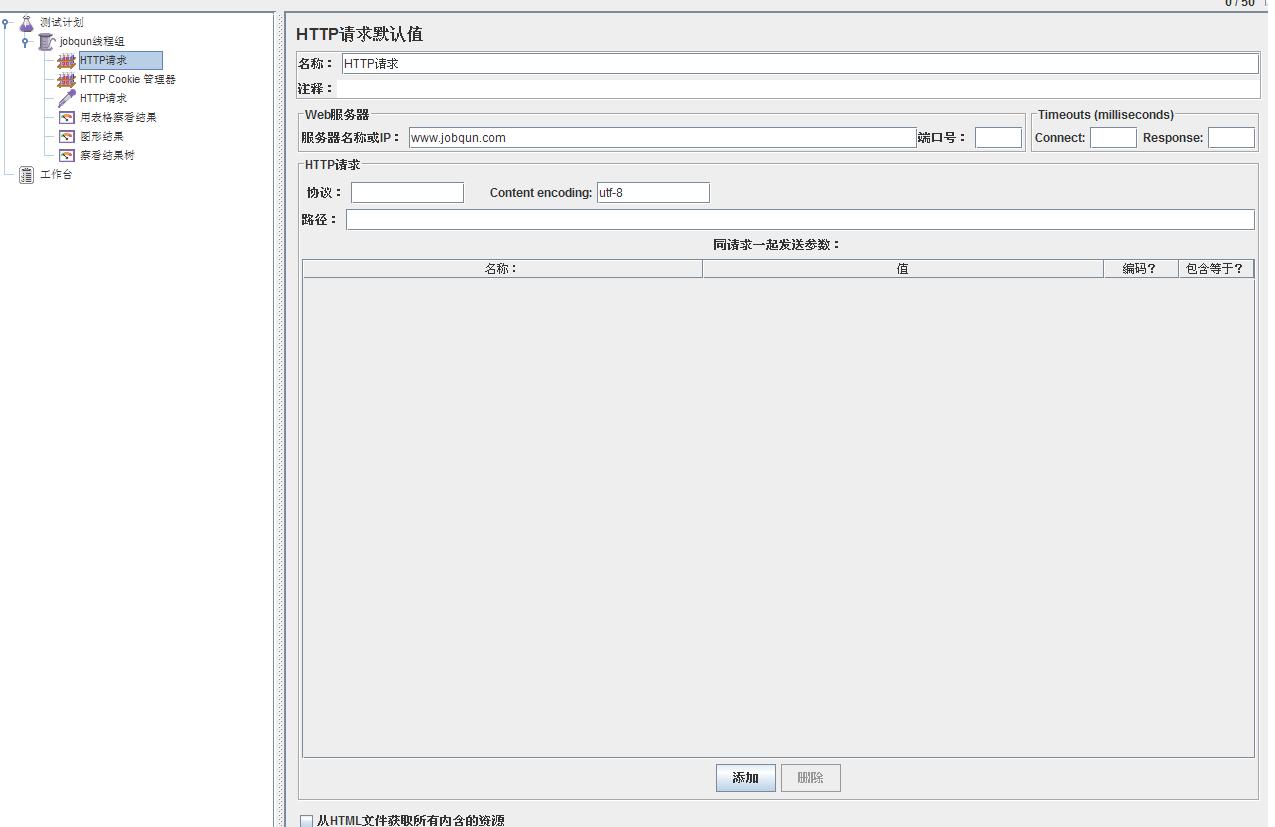
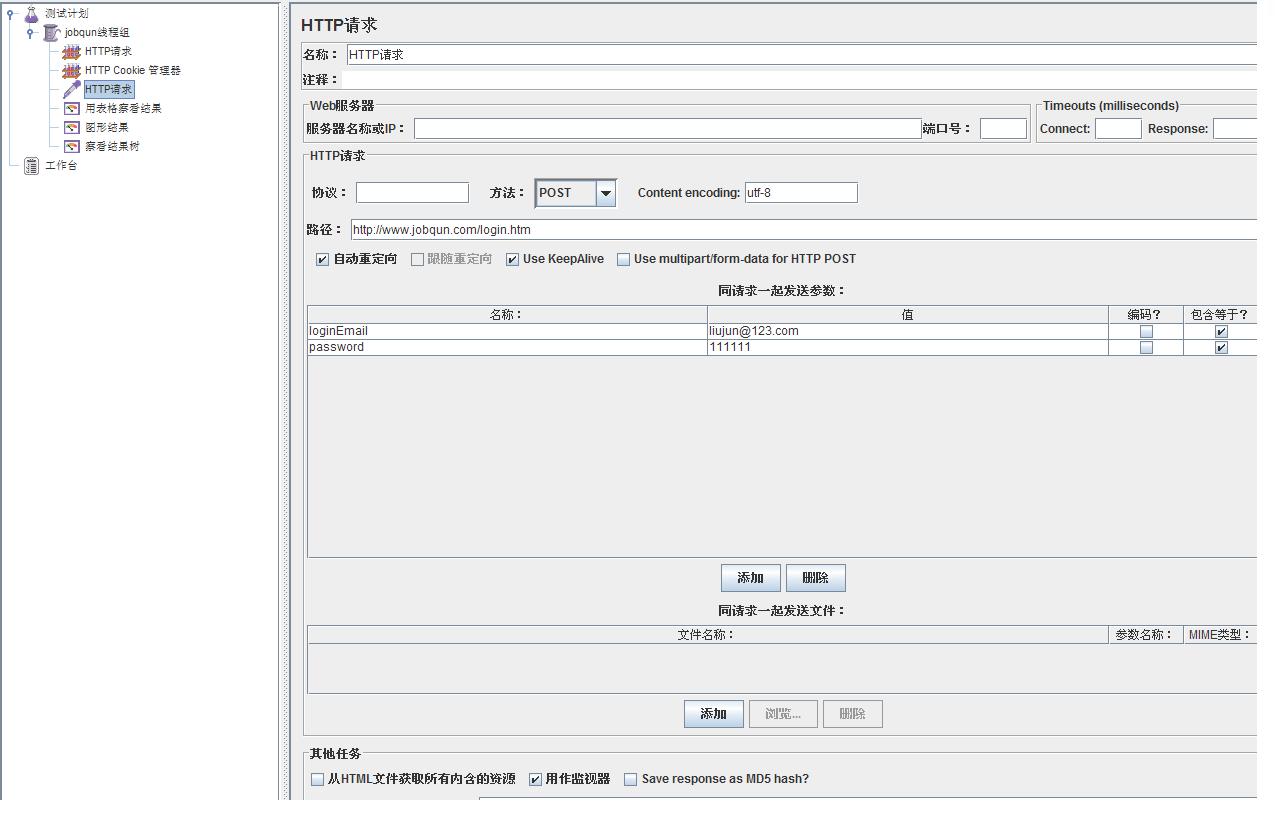
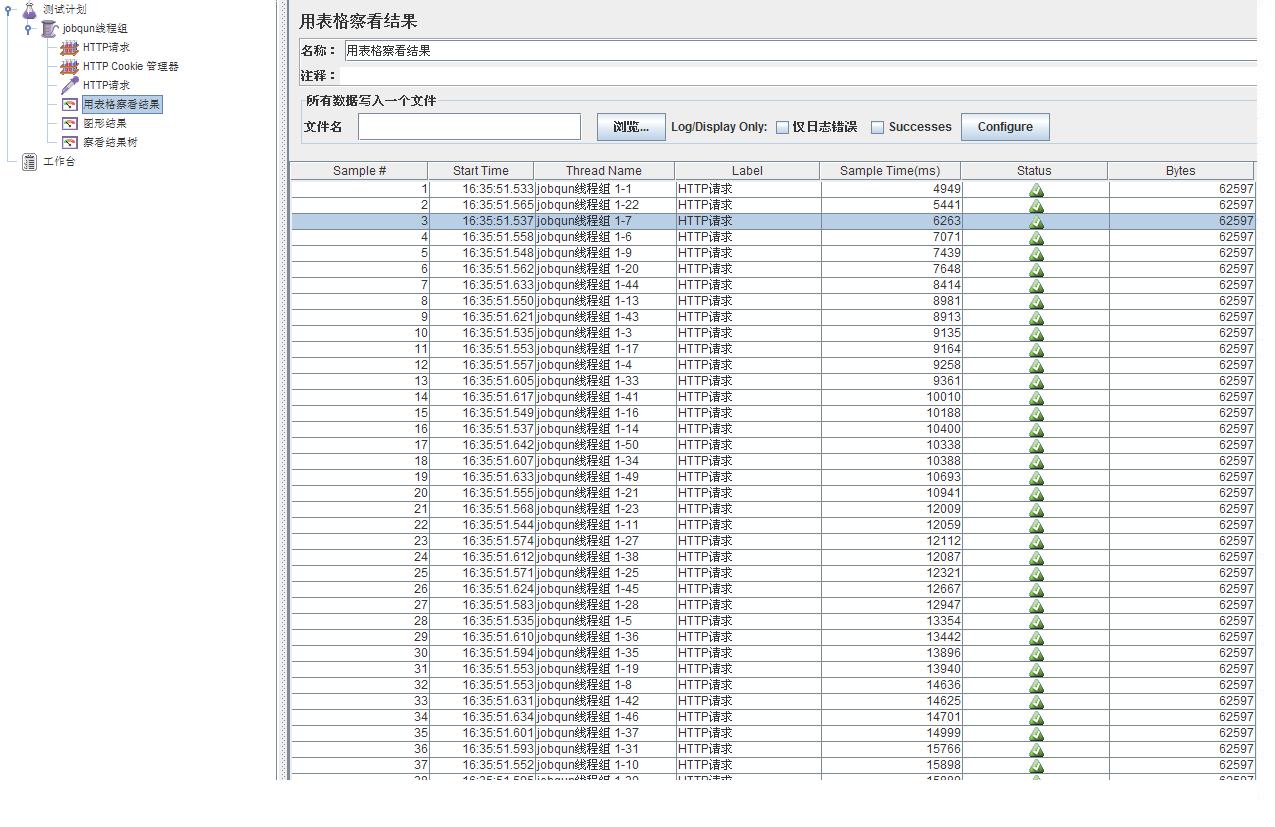
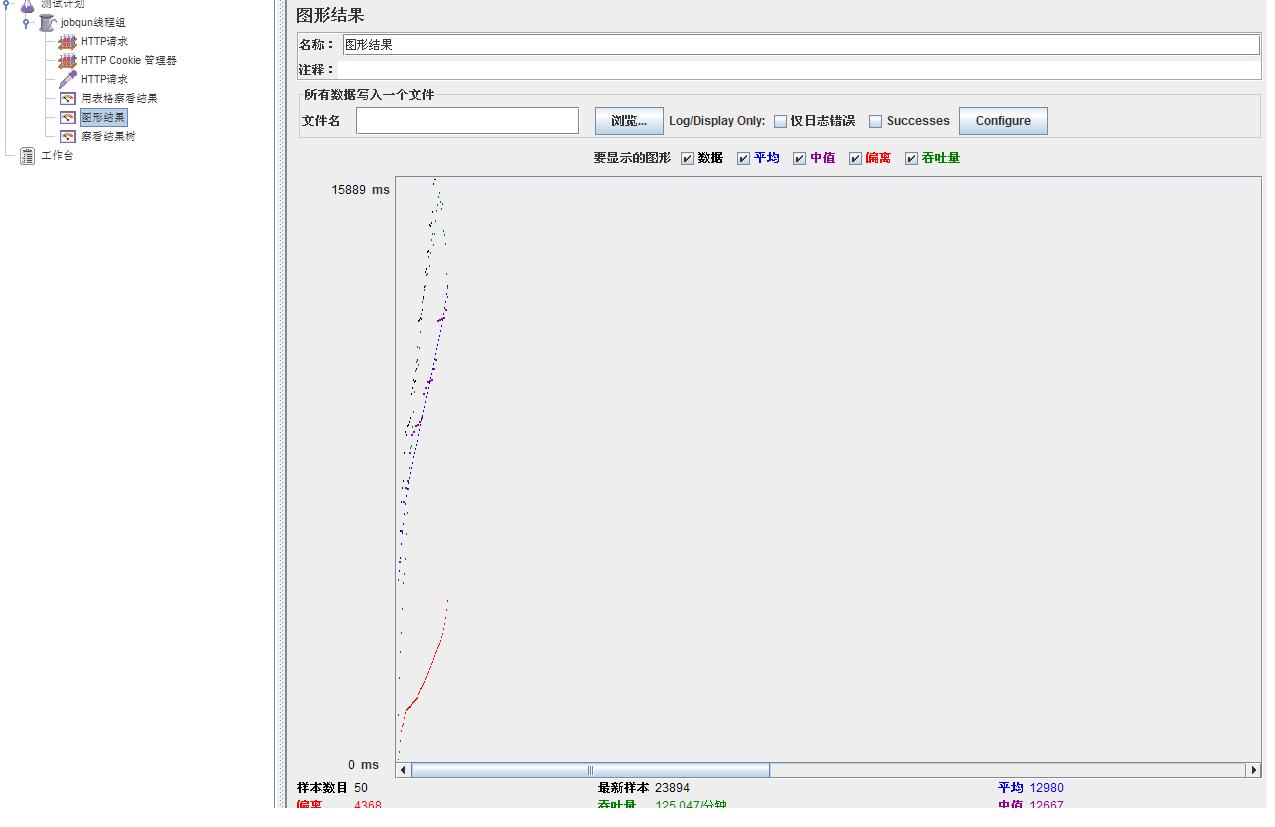
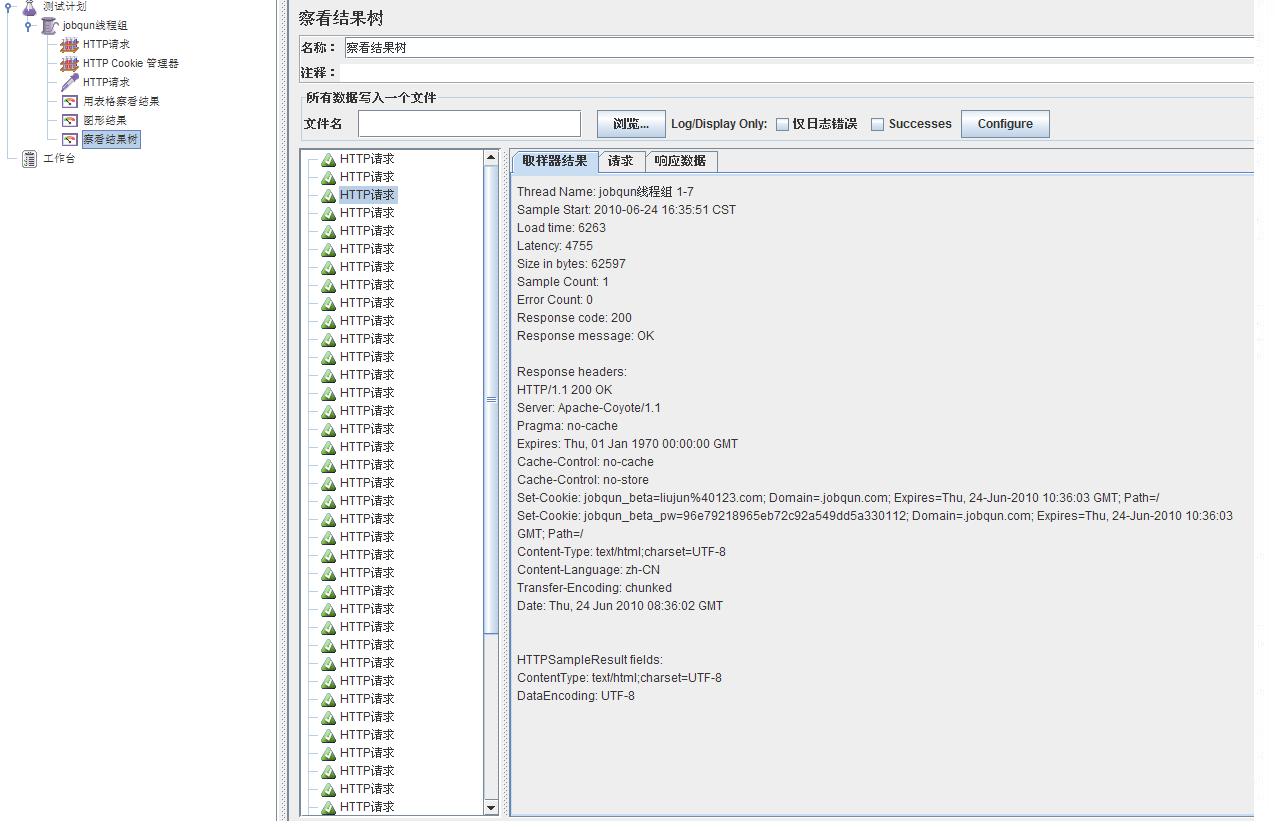
　　这个方法验证了我们刚才这个猜想。在完成通过MAVEN启动Jmeter的分析过后，我们所要做的事情就是如何解析Jmeter运行后所得到的测试结果，并将这个测试结果以相关的格式展现出来。

[**jmeter 压力测试**](http://skly-java.iteye.com/blog/1682031)

**博客分类：**

* [测试](http://skly-java.iteye.com/category/248624)

原文：<http://rentianchou.iteye.com/blog/698561>

今天初次用了下apache的jmeter测试工具测试网站的压力测试，步骤如下：  
先去apache官网下了最新的2.3.4版本，下下来直接解压缩到任意位置就可以直接使用了(确定机器上安装了JDK)，然后找到bin目录下的jmeter.bat文件，双击即运行出测试界面.  
右击菜单添加线程组，如图1，  
  
 并配置好参数如图2  
  
 。线程数代表有多少个线程，也就是代表多少个用户；Ramp-Up Period(in-seconds)代表隔多长时间执行，0代表同时并发;循环次数就是代表执行几次。   
然后在线程组上右击 添加Http请求默认值，它在这里类似于一个全局的，以后再配置类似HttpRequest的某些东西就不要再配置了，如图3为配置参数  
  
   服务器名称或ip配置为服务器的ip或者域名都可以  如果是tomcat的话  端口号为测试tomcat的端口号 content encoding为请求的编码格式  
然后右击线程组添加http请求并配置好参数 如图4  
  
  这里的服务器名称和ip就不用再配置了  因为前面已经配置好全局的了。  然后把需要的参数添加进去，这里我测试的是登陆，所以添加了2个参数 一个loginEmail = [liujun@123.com](mailto:liujun@123.com) password = 111111。  
然后右击线程组 依次添加用图表格查看结果、图形结果、查看接过树。  
最后ctr+R ，我们查看表格查看结果界面如图5  
  
 ，  同理 图6  
  
   图7  


下面是一些测试中的一些术语解释  
1.1       Aggregate Report  
Aggregate Report是 JMeter 常用的一个 Listener，中文被翻译为“聚合报告”。例如只有一个登录的请求，那么在Aggregate Report中，会显示一行数据，共有10个字段，含义分别如下。  
Label：每个 JMeter 的 element（例如 HTTP Request）都有一个 Name 属性，这里显示的就是 Name 属性的值  
#Samples：表示你这次测试中一共发出了多少个请求，如果模拟10个用户，每个用户迭代10次，那么这里显示100  
Average：平均响应时间——默认情况下是单个 Request 的平均响应时间Median：中位数，也就是 50％ 用户的响应时间(ms)  
90% Line：90％用户的响应时间（ms）  
Min：最小响应时间  
Max：最大响应时间  
Error%：本次测试中出现错误的请求的数量/请求的总数  
Throughput：吞吐量——默认情况下表示每秒完成的请求数（Request per Second）  
KB/Sec：每秒从服务器端接收到的数据量  
1.2       用表格察看结果  
样本数目：总共发送到服务器的请求数  
最新样本：代表时间的数字,是服务器响应最后一个请求的时间。  
平均值：总运行时间除以发送到服务器的请求数。  
偏离：服务器响应时间变化、离散程度测量值的大小，或者，换句话说，就是数据的分布。  
1.3       图形报表  
样本数目：总共发送到服务器的请求数。  
最新样本：代表时间的数字,是服务器响应最后一个请求的时间。  
吞吐量：服务器每分钟处理的请求数。  
平均值：总运行时间除以发送到服务器的请求数。  
中间值：时间的数字，有一半的服务器响应时间低于该值而另一半高于该值。  
偏离：服务器响应时间变化、离散程度测量值的大小，或者，换句话说，就是数据的分布。

Jmeter不单单可以压力测试，还可以jdbc测试、ftp、java测试、XML等好多功能。

### [Jmeter：图形界面压力测试工具](http://skly-java.iteye.com/blog/1681985)

**博客分类：**

* [测试](http://skly-java.iteye.com/category/248624)

原文：<http://www.cnblogs.com/cotty/archive/2012/08/30/2663562.html>

Jmeter是一款强大的图形界面压力测试工具，完全用Java写成，关于Jmeter的介绍，网上其实有不少的文章，我原本是不想再重复写类似文章的，但我发现有些很关键性的，在我们测试中一定会用到的一些设置或操作很少见到有文章写清楚的，比如有这样的一个常见问题：如何对一个网站的多个链接进行压力测试？因为通常情况下，我们要测试的不是一个链接，假如我们测试的是某一网站的话，而这些链接我们通常都是写入到一个文本文件中，测试的过程中，会随机读取文件中的链接进行压力测试，我们所熟知的一些压力测试工具，例如[Siege](http://www.cnblogs.com/2011/01/27/2737)就可以这样干。

在我使用Jmeter的过程中，我发现我周围有之前使用过Jmeter的同事，但即使如此，他们对如何将多个链接写入到文本文件中并随机读取进行压力测试也不清楚，故我打算还是在有空的时候写一篇这样的文章，以方便使用者。

**一、 官方网址**

<http://jakarta.apache.org/jmeter/>

**二、 运行**

下载解压后，在目录jakarta-jmeter-2.4\bin下可以见到一个jmeter.bat文件，双击此文件，打开初始界面如下图：



注意的是上面显示的是中文，如果你想使用其他语言，比如英文，那么通过菜单选项->选择语言->英文即可，当然转为中文也是同样操作。

**三、 运行预准备**

我们现在要对子猴博客来进行一番压力测试，压力测试对象为随机的几个网页链接，这几个链接是写在一个文本文件中的，在压力测试的时候会随机读取。

1、  建立一个线程组，如下图



为什么要建立线程组？原因很简单，因为我们要模拟多个线程（用户）来访问网站。

线程组创建界面如下图：

名称可以随意填写，默认选中“继续”即可。

线程属性部分中，线程数是启动多少个线程，我这里填写的是60，Ramp-Up Period (in seconds)表示线程之间间隔多少时间允许，单位是秒，比如如果填写120，那么120/60=2表示60个线程间每隔2秒钟请求网站。

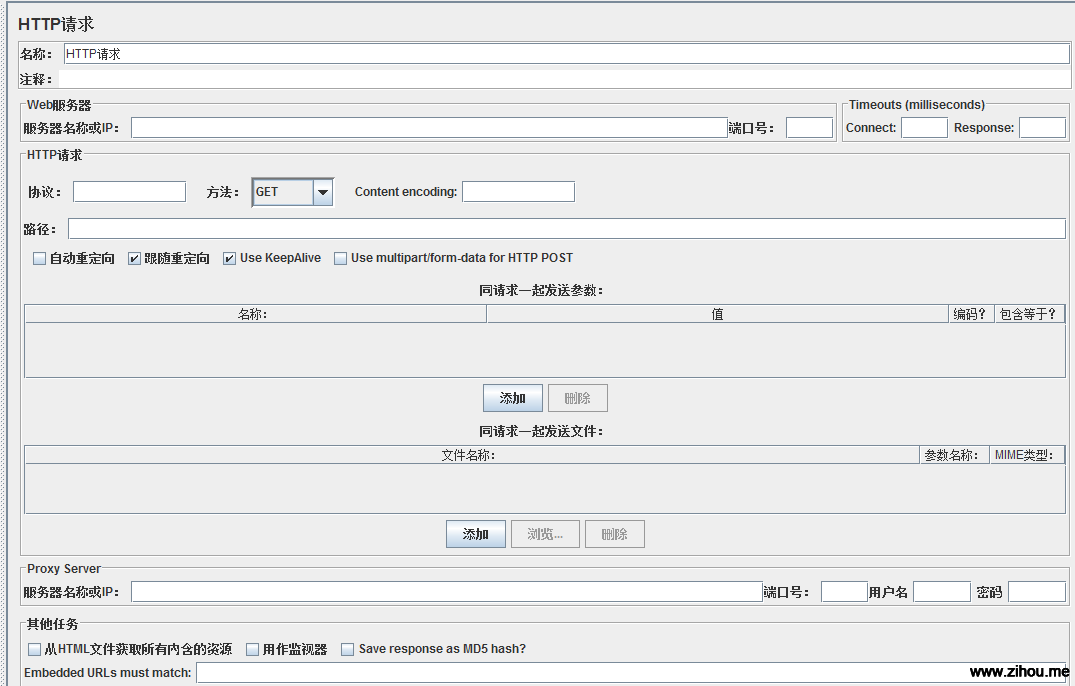
循环次数：60个线程运行完毕算是一次，循环次数就是这样的一个请求过程运行多少次，我这里填写的是1000.

每次修改一个设置后，别忘记了保存一下。

2、  设置请求服务器、压力链接等信息

接下来很自然的是，我们要测试的网站地址是什么？链接是什么？所以现在我们就来设置这些信息。

右键点击我们刚创建的线程组，在弹出的菜单中，选择添加->Sampler->Http请求，弹出如下图界面：



名称：随意填写

注释：可有可无

服务器名称或IP：我这里填写 zihou.me

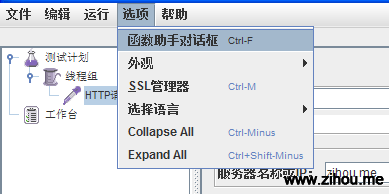
端口号：我这里填写80

Timesout部分可以不填

HTTP请求部分的协议：http，方法我选择的是GET，Content encoding我填的是UTF-8.

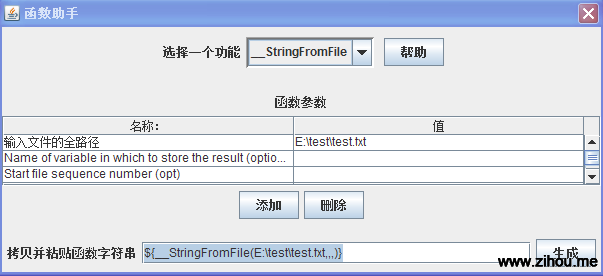
**路径**：

这里值得一提的就是这个路径，假如你只是对一个链接进行压力测试，那么这里就非常简单了，你就直接填写一个链接就够了，比如http://www.zihou.me，但很多情况下都不是这样的，我们这里需要多个链接，就如同刚开始讲到的那样，我们要将多个链接保存到一个文本文件中，然后随机读取进行压力测试。我们可以这么干，如图：

打开界面如下图：



在选择一个功能下拉列表中选择\_StringFromFile，然后在本机新建一个测试文件text.txt，在第一行（你也可以不在第一行）的值中填写测试文件的路径，如下图：



然后点击“生成”按钮，在生成按钮的左边文本框中将生成一个字符串如：

${\_\_StringFromFile(E:\test\test.txt,,,)}

在测试文件中，我们每行写一个URL链接，如下格式：

2011/03/26/3054

2011/03/26/3052

2011/03/26/3042

2011/03/25/3040

2011/03/25/3034

2011/03/24/3027

注意，每行前面并没有http://www.zihou.me这样的信息，因为我们在前面已经填写了服务器地址为zihou.me，这里就没必要再为每个url填写这个相同前缀了；另外，上面的url格式也只是个例子，表示域名后的部分，但很多时候url后面跟的都是各参数，比如http://www.zihou.me/p?a=1&b=3，如果是这种情况，则上面的链接可以写为：

a=1&b=3这样的形式。

现在我们可以来填写前面所说的路径了，如下：

/ ${\_\_StringFromFile(E:\test\test.txt,,,)}

这样一来，当我们并非请求的时候，就会从test.txt中随机选择url来进行压力测试。

另外值得注意的一个地方是，如果参数中有中文的情况，运行的时候可能会出现乱码，这个时候就需要注意你在Jmeter中的编码设置与你要请求的网页编码是一致的。

路径文本框下面的选项，可以按默认的就成，Use multipart/form-data for HTTP POST是当请求中有附件的情况，一般情况下都不用选中的。

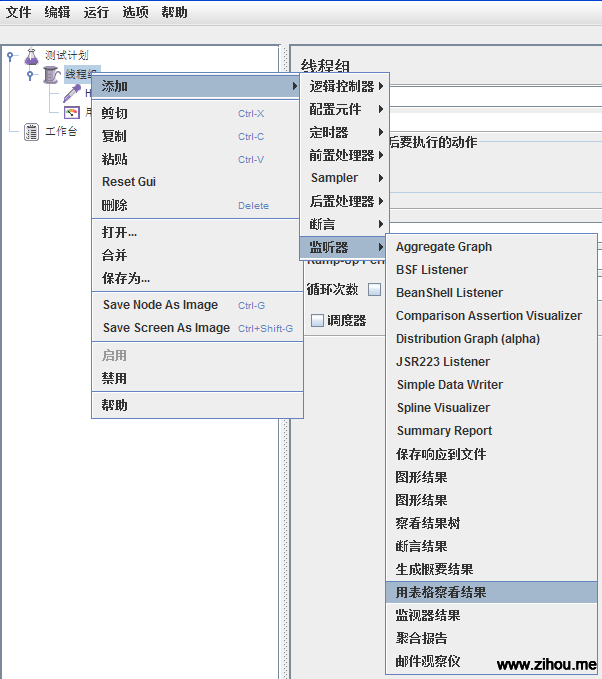
**同请求一起发送参数部分**：

如果你仅仅是对某一个固定的页面进行压力测试，那你就可以不用进行上面所说的在文本文件中设置URL的过程了，直接在这里设置就行了，这里的名称就是参数名，值就是参数值，在这里添加就OK了，但你如果是压力多个链接，那这里你可以不用填。

3、查看运行结果

上面设置好后，接下来很自然的想到如何查看运行结果呢？Jmeter在这方面提供了好些个查看方式，有表格形式，有曲线形式等等，我个人认为查看表格形式就足够了！

鼠标右键点击线程组，在弹出的菜单中选择添加->监听器->用表格查询结果，如下图：



弹出界面如下图：

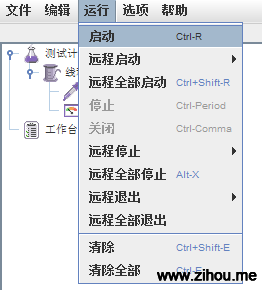


当然，你也可以选择其他查看方式，你也可以看到在监听器菜单中，有好多种方式。

**四、 运行**

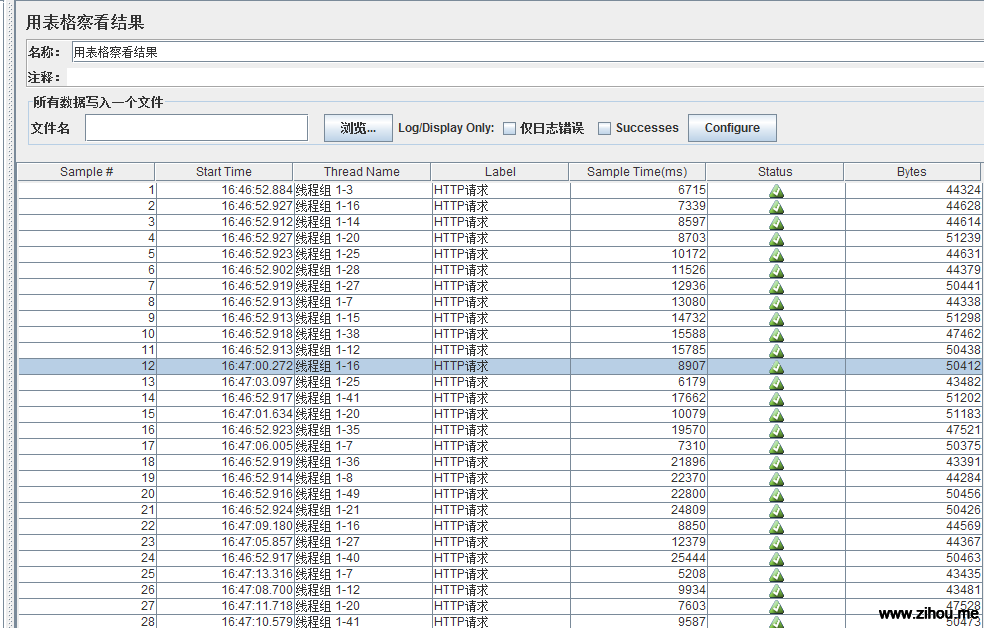
好了，现在我们终于可以运行了！

选择菜单项运行->启动，如下图：



当然，在运行前，你得把所有修改都保存了。

运行后的结果表格如下：



各属性如下：

Sample：每个请求的序号

Start Time：每个请求开始时间

Thread Name：每个线程的名称

Label：Http请求名称

Sample Time：每个请求所花时间，单位毫秒

Status：请求状态，如果为勾则表示成功，如果为叉表示失败。

Bytes：请求的字节数

如果Status为叉，那很显然请求是失败了，但如果是勾，也并不能认为请求就一定完全成功了，因为还得看Bytes的字节数是否是所请求网页的正常大小值，如果不是则说明发生了丢包现象，也不是完全成功。

在下面还有几个参数

样本数目：也就是上面所说的请求个数，成功的情况下等于你设定的并发数目乘以循环次数。

平均：每个线程请求的平均时间

最新样本：表示服务器响应最后一个请求的时间

偏离：服务器响应时间变化、离散程度测量值的大小，或者，换句话说，就是数据的分布（这个我不是很理解）。

在上面的参数中，我个人认为只要看Status和Bytes这两个就够了。

好了，主要的就介绍这么多了，其实Jmeter做压力测试的范围远远不止于对Web，它还可以用于Java小服务程序、CGI 脚本、Java 对象、数据库， FTP 服务器等等，更多的使用可以根据自己的需要去研究掌握。

最后还提到的一点是：你一定注意到了，HTTP请求和结果查看都是在线程组这一节点下创建的，其实，你也可以不用一定要在线程组下创建，你在测试计划下建立也可以，但创建在线程组下面可以使得我们的测试意图一目了然，也就是我们的HTTP请求和结果查看都是基于我们设定的线程组的。

[**使用JMeter创建数据库（Mysql）测试**](http://skly-java.iteye.com/blog/1682172)

**博客分类：**

* [测试](http://skly-java.iteye.com/category/248624)

[软件测试](http://www.iteye.com/blogs/tag/%E8%BD%AF%E4%BB%B6%E6%B5%8B%E8%AF%95)

原文：<http://www.cnblogs.com/fnng/archive/2011/07/26/2117600.html>

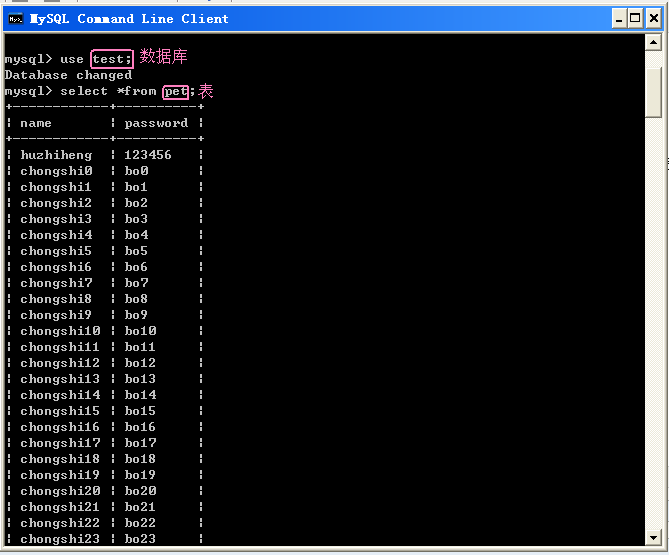
我的环境：MySQL：mysql-essential-5.1.51-win32

　　　　　jdbc驱动：我已经上传到csdn上一个：[http://download.csdn.net/source/3451945](http://download.csdn.net/source/3451945" \t "_blank)

　　　　　JMeter：jmeter-2.4  任意版本都行。

1.首先我们要有一个可以做测试的数据库，当然，里面要有数据，不然怎么测呢？我的上一篇文章《**[Eclipse连接MySQL数据库（傻瓜篇）](http://www.cnblogs.com/fnng/archive/2011/07/18/2110023.html)**》，里面教你如何通过程序批量插入数据。

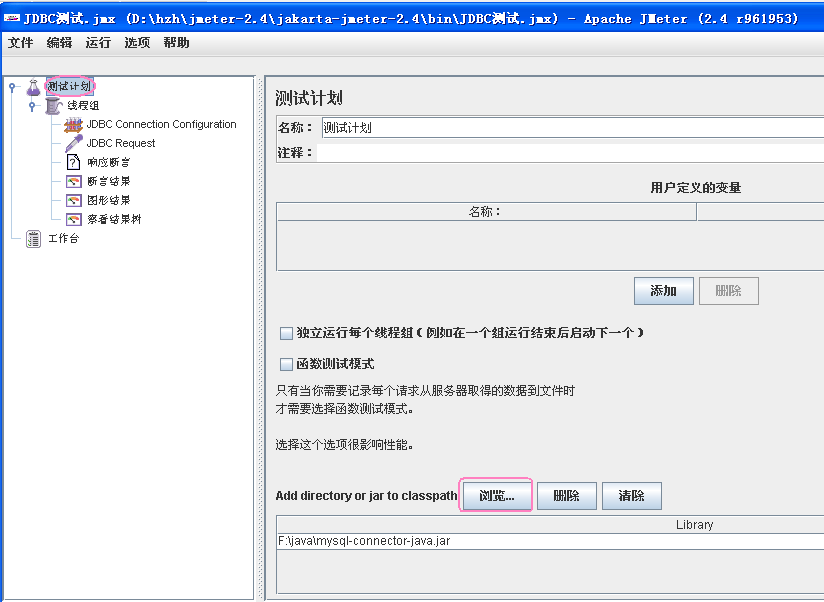
来看一下我的数据：



我可能有点啰嗦 :）  不管你用什么方式，你的数据库里要有点数据。

2.

打开JMeter,点击测试计划，



点击“浏览...”按钮，将你的JDBC驱动添加进来。

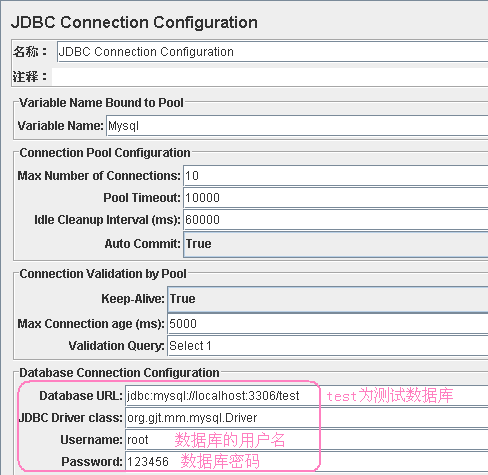
3.

添加一个线程组，

右键点击“线程组”，在下面添加一个“JDBC Connection Configuration”

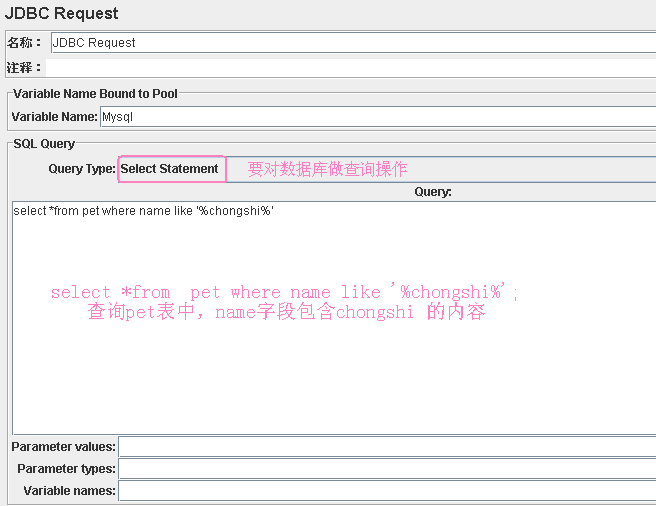


来配置一下JDBC Connection Configuration页面。



4.

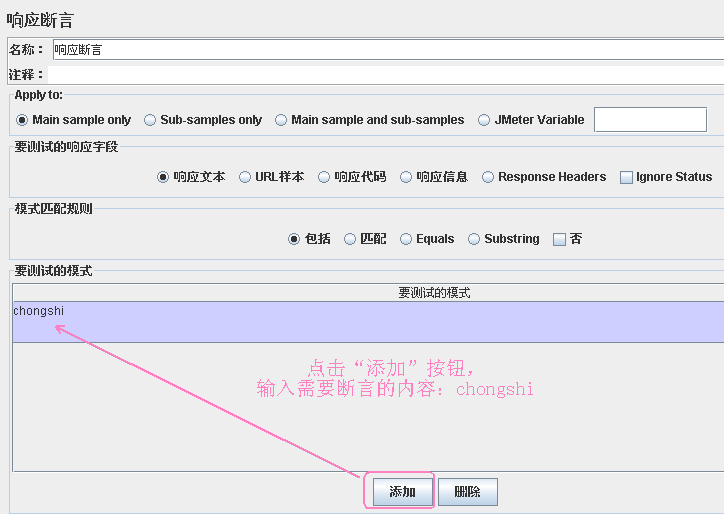
右键点击“线程组”，在下面添加一个“JDBC request”



5.

添加断言。

右键点击线程组---->添加--->断言---->响应断言。



6.

我们来添加一些监听器来行查看

添加一个断言结果：

右键点击线程组---->添加--->监听器---->结果断言。

添加一个图形结果：

右键点击线程组---->添加--->监听器---->图形结果。

添加一个查看结果树：

右键点击线程组---->添加--->监听器---->查看结果树。

下面是添加所有东东的列表：

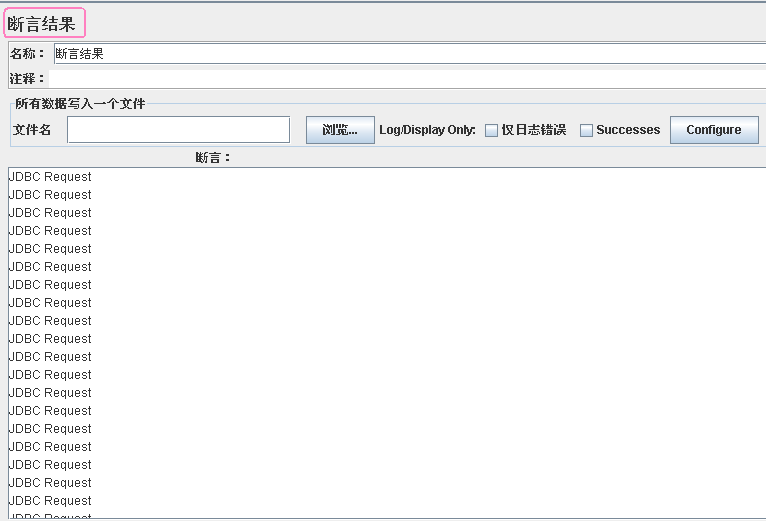


7.

在线程组页面设置用户数、启动时间、循环次数

点击菜单栏“运行”----“启动”

下面是结果：



# http接口测试——Jmeter接口测试实例讲解

发表于7个月前(2015-09-29 19:00)   阅读（9606） | 评论（[3](http://my.oschina.net/hellotest/blog/512482" \l "comments)） 154人收藏此文章, [我要收藏](javascript:add_to_favor(512482,3))

赞3

[**4月23日，武汉源创会火热报名中，期待您的参与>>>>>**](http://city.oschina.net/wuhan/event/2160100)   http://my.oschina.net/img/hot3.png

摘要 最近做的项目需要测试很多接口，上网查一查，发现完整讲述接口测试的资料太少，所以最近自己做完这个项目，把测试的东西整理一下和大家分享一下，希望对看到的人有所帮助

**一、测试需求描述**

   1、 本次测试的接口为http服务端接口

   2、 接口的主要分成两类，一类提供给查询功能接口，一类提供保存数据功能接口，这里我们举例2个保存数据的接口，因为这两个接口有关联性，比较有代表性；

    3、接口描述：

**保存信用卡账户信息接口**：

            传入参数：

                args={

                    "clientNo":"434343556",

                    "alias": "\*\*信用卡2",

                    "cardName": "长城\*\*\*\*\*卡2",

                     "cardNo": "25622356788251",

                    }

            传出参数：

                    保存成功：{"returnCode":"0","returnMsg":"保存成功"}

                    保存失败：{"returnCode":"1","returnMsg":"保存失败"}

            保存逻辑：数据传入进来，验证通过，保存到信用卡账户表中

**保存信用卡账单接口：**

            传入参数：

                args={

                    "clientNo":"434343556",

                    "accountName": "测试",

                    "billDate": "08",

                    "billMonth": "201509",

                     "cardNo": "25622356788251",

                    "currentPayment": "欠款459.80",

                    "paymentDate": "2015-09-25 09:00:00",

                    }

            传出参数：

                    保存成功：{"returnCode":"0","returnMsg":"保存成功"}

                    保存失败：{"returnCode":"1","returnMsg":"保存失败"}

                     保存逻辑：保存时先去信用卡信息表查看clientNo对应的表是否存在，如存在则数据校验通过，

                                       将数据保存进入信用卡账单表

**注**：这个保存逻辑在接口开发设计文档中可能没有写或写的不详细，这时要与开发接口人员或产品人员多多沟通去熟悉接口逻辑

**二、测试分析**

    1、**逻辑分析**

        1)、从保存逻辑上来看，这两个接口明显是有依赖关系的，所以我们先测试信用卡账户信息接口，再测试保存信用卡账单接口

        2)、接口传入的数据，最终是保存到数据库中，所以当接口返回保存成功的时候，我们也要去对应的数据库表中核对相应的数据(这里可以用jmeter链接数据库进行操作代理手工)

        3)、当出现保存失败的情况时，我们需要查看系统的日志，所以我们也要准备好查看日志的权限和地址

**2、测试工具的准备**

        1)、单个接口测试，我们使用火狐的插件poster

        2)、多个接口测试，我们使用Jmeter进行测试

**三、使用工具测试**

    1、Poster工具的使用方法见

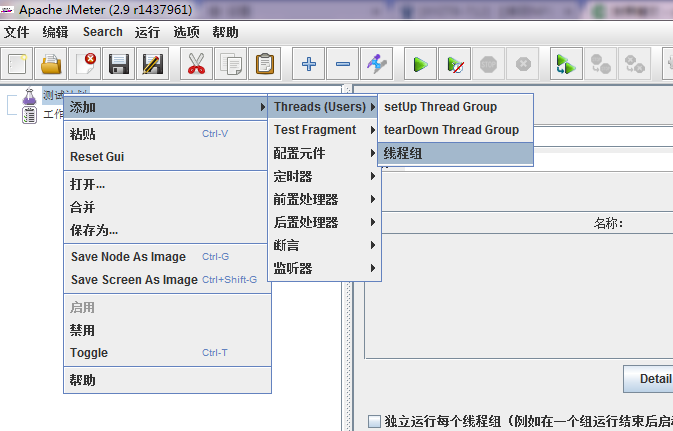
<http://note.youdao.com/share/?id=a3efa149d165258710ebf1bd04079f72&type=note>

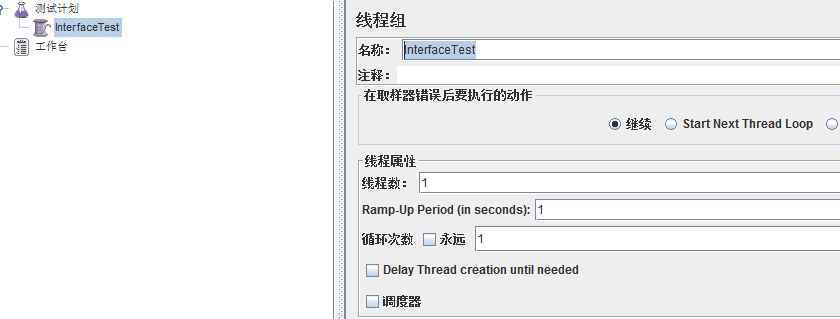
    2、使用Jmeter对接口测试

            首先我们说一下为什么用Poster测试后我们还要用Jmeter做接口测试，在用poster测试时候会发现的是一个接口一个接口的测试，我们每次测试 成功后的数据，在工具中是无法保存的，再次测试的时候我们还要重新输入测试的数据，当我们测试一个接口的时候可能感觉不明显，但是当你测试几十个接口的时 候，你就会发现使用Jmeter的好处，如果按测试阶段来说冒烟测试我们用poster，集成测试我们用Jmeter

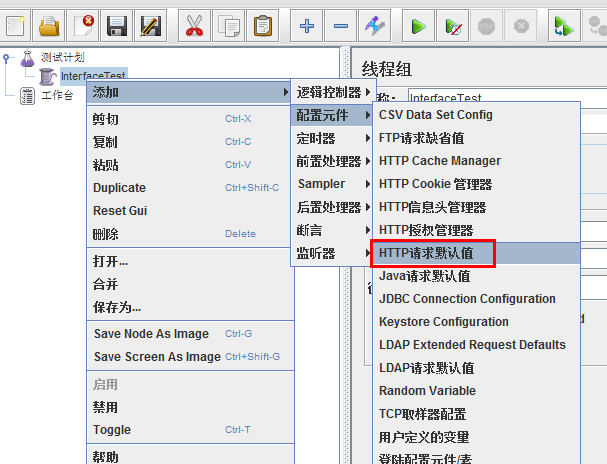
**四、使用Jmeter接口测试**

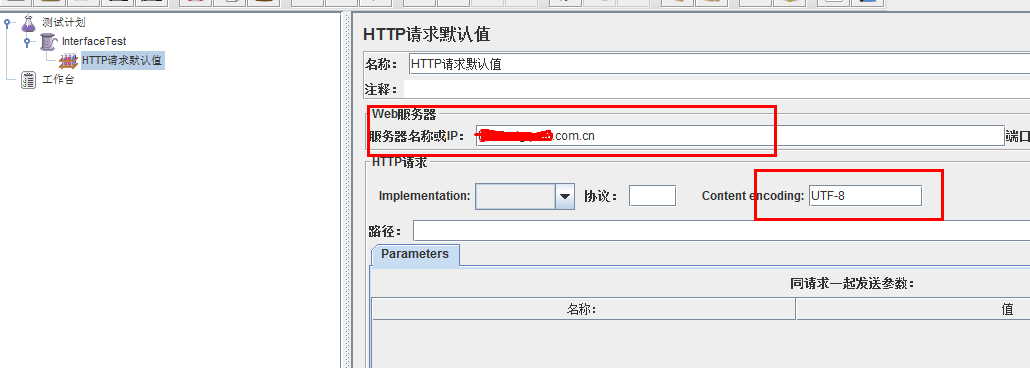
        1、首先邮件添加一个线程组，这里我们重命名InterfaceTest

[](http://static.oschina.net/uploads/space/2015/0929/185303_67iu_1391889.png)

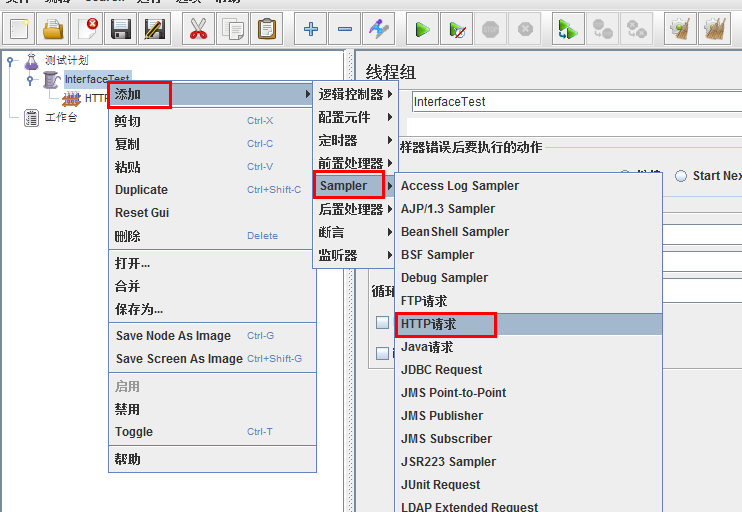
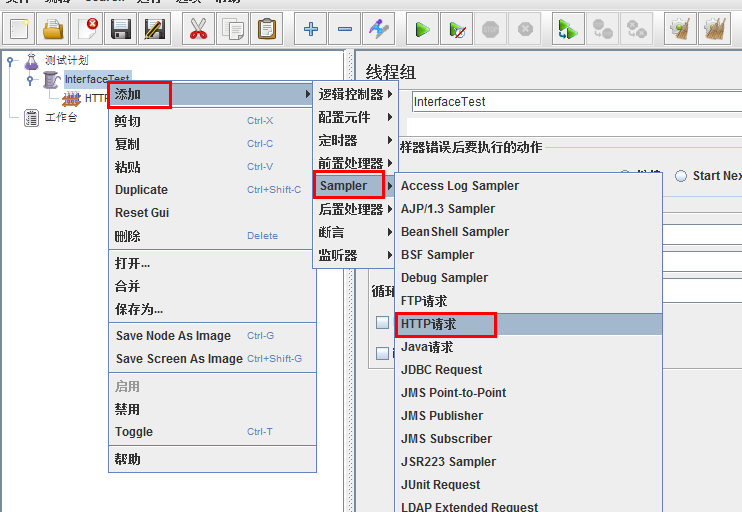
[](http://static.oschina.net/uploads/space/2015/0929/185339_6ids_1391889.png)

        2、在线程组上添加一个Http默认请求，并配置服务器的IP地址和传输编码

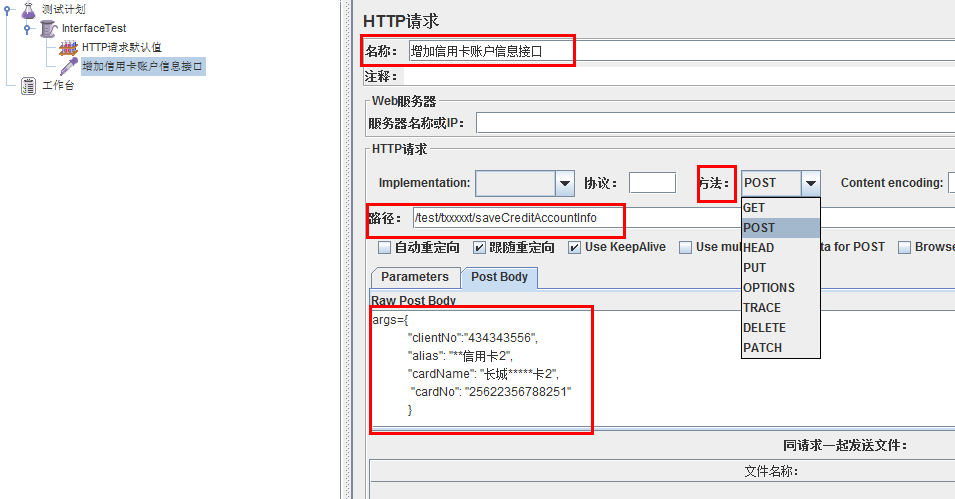
[](http://static.oschina.net/uploads/space/2015/0929/185441_r3Jk_1391889.png)

[](http://static.oschina.net/uploads/space/2015/0929/185441_a4M7_1391889.png)

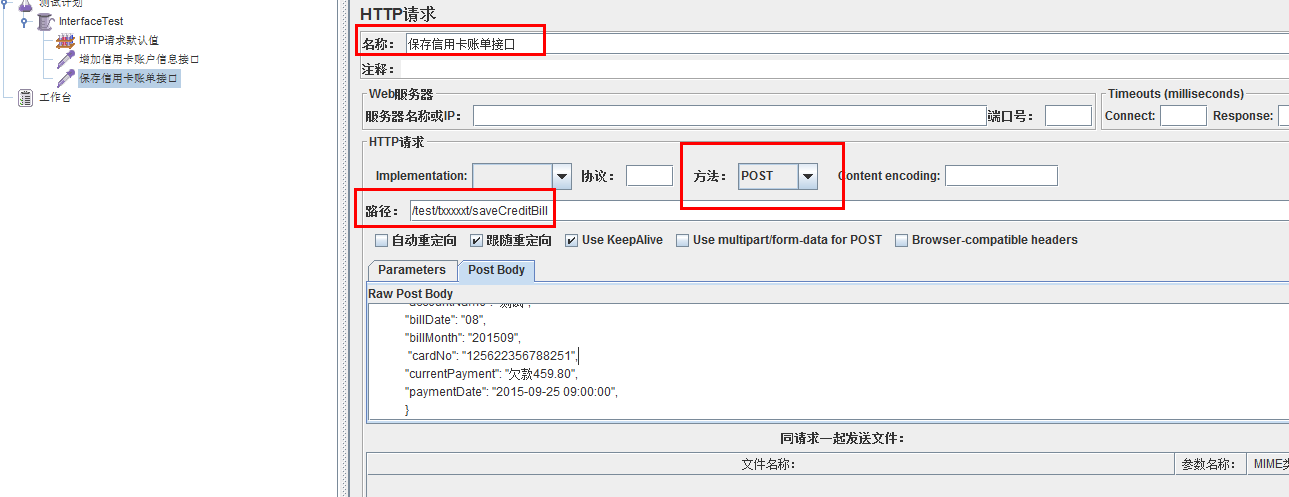
        3、在线程组中添加一个HTTP请求，这里我们重命名“增加信用卡账户信息接口”

[](http://static.oschina.net/uploads/space/2015/0929/185457_bVhj_1391889.png)    [](http://static.oschina.net/uploads/space/2015/0929/185600_kuJ2_1391889.png)

        4、配置接口请求信息，这配置示例如下：

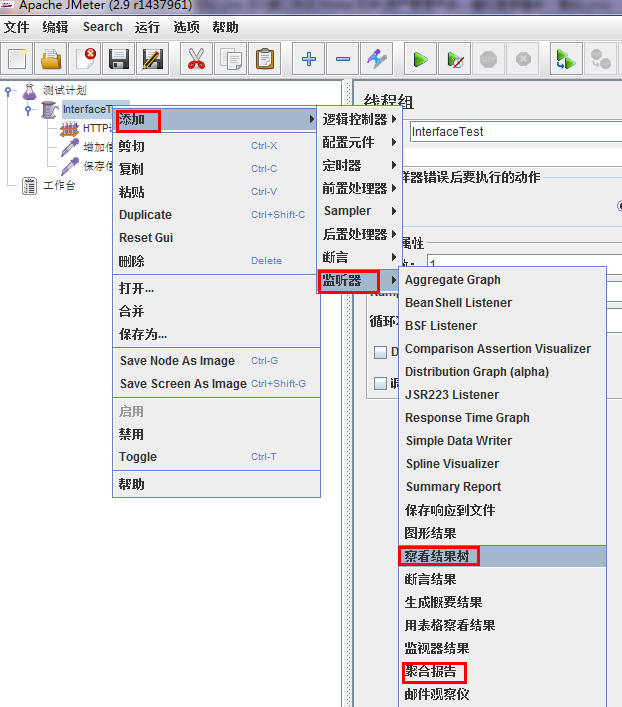
[](http://static.oschina.net/uploads/space/2015/0929/185631_lFg8_1391889.png)

        5，在保存信用卡账单接口请求，示例如下：

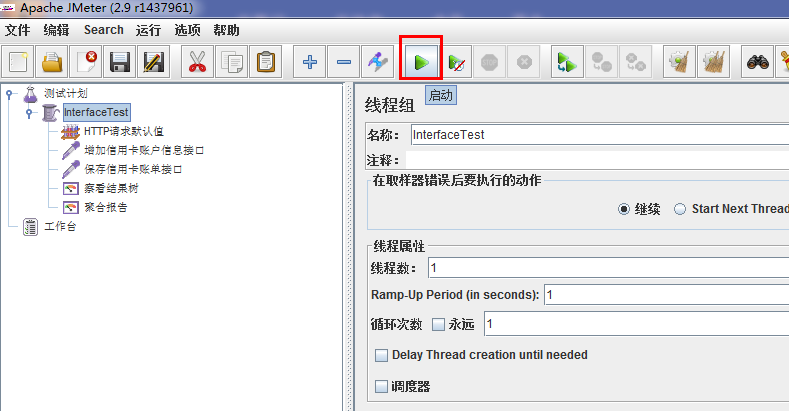
[](http://static.oschina.net/uploads/space/2015/0929/185709_HosX_1391889.png)

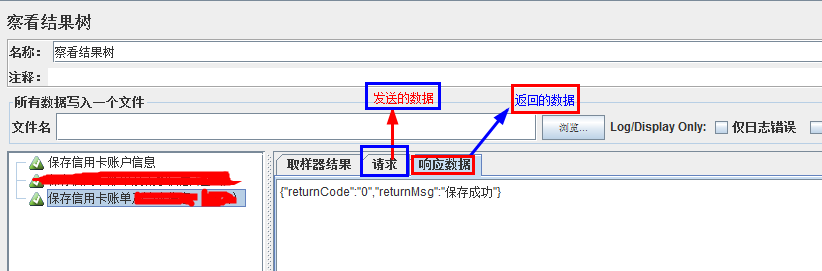
        注：由于Jmeter请求线程组内的请求时从第一个开始执行，所以我们将需要最先执行的请求放在前面

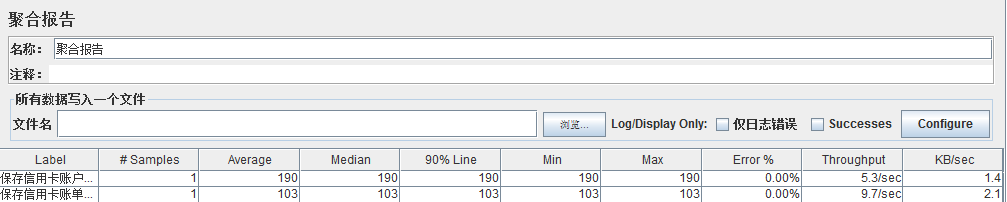
        6、在线程组上添加监听器，察看结果树和聚合报告

[](http://static.oschina.net/uploads/space/2015/0929/185737_GpUx_1391889.png)

        7、点击启动，运行结束后查看，结果树和聚合报告

[](http://static.oschina.net/uploads/space/2015/0929/185824_Vuru_1391889.png)

[](http://static.oschina.net/uploads/space/2015/0929/185908_BDCE_1391889.png)

[](http://static.oschina.net/uploads/space/2015/0929/185908_P1vT_1391889.png)

        8、去数据库中核对数据

        9、大批量数据制造

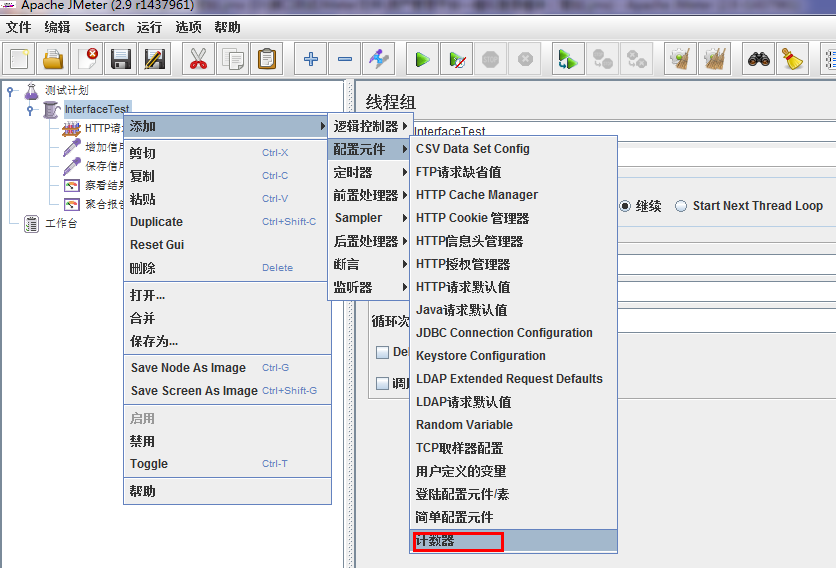
            思路：

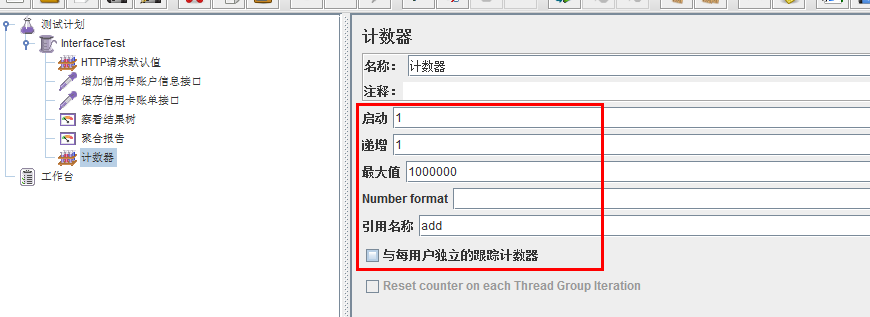
        1)、可参数化的参数，保存信用卡账户信息接口(clientNo,cardNo），保存信用卡账单接口(clientNo,cardNo,billMonth，paymentDate）

        2)、两个接口的依赖关系，保存信用卡账单接口(clientNo,cardNo）要和信用卡账户信息接口(clientNo,cardNo）的两个相同，也就是说这两个要用一个参数，且还不能重复

        根据上面两个特点，(clientNo,cardNo）我们选取使用计数器，每循环一次计数器加1，那么我们将线程组设置循环执行1万次；billMonth，paymentDate，这两个日期我们是使用随机函数${\_\_Random(1,9,)}，将月份参数化；

        3)、在线程组上创建计数器，配置如下：

[](http://static.oschina.net/uploads/space/2015/0929/185958_jLXw_1391889.png)

[](http://static.oschina.net/uploads/space/2015/0929/185958_y1ft_1391889.png)

        4)、引用计数器和随机函数

        信用卡账户接口传入参数

                args={

                    "clientNo":"${add}434343556",

                    "alias": "\*\*信用卡2",

                    "cardName": "长城\*\*\*\*\*卡2",

                     "cardNo": "${add}25622356788251",

                    }

        账单接口传入参数

        args={

                    "clientNo":"${add}434343556",

                    "accountName": "测试",

                    "billDate": "08",

                    "billMonth": "20150${\_\_Random(1,9,)}",

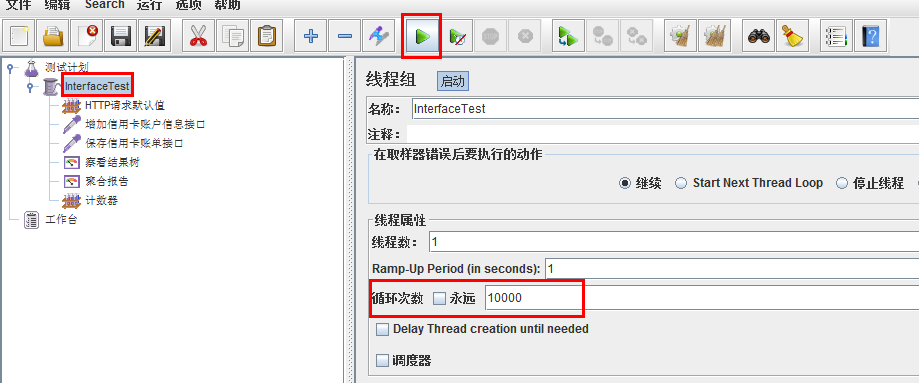
                     "cardNo": "${add}25622356788251",

                    "currentPayment": "欠款459.80",

                    "paymentDate": "2015-0${\_\_Random(1,9,)}-25 09:00:00",

                    }

        5)、设置线程组循环测试，点击运行,查看运行结果，再去数据库看看，大批量数据就这样产生了

[](http://static.oschina.net/uploads/space/2015/0929/190034_WoC7_1391889.png)