

# 性能测试工具——Jmeter

## 一、Jmeter 简介

### Jmeter 是啥？

Apache JMeter 是 Apache 组织的开放源代码项目，是一个纯 Java 桌面应用，用于压力测试和性能测量。它最初被设计用于 Web 应用测试但后来扩展到其它测试领域。

### Jmeter 有啥用？？

Apache JMeter 可以用于对静态的和动态的资源(文件, Servlet, Perl 脚本, Java 对象, 数据库和查询, FTP 服务器或是其它资源)的性能进行测试。JMeter 可以用于分析不同压力条件下的总体性能情况。也可以使用 JMeter 提供的图形化界面，分析性能指标或者在高负载情况下测试你的服务器,脚本,对象。

### Jmeter 与 LR 有啥区别？？？

	Loadrunner	Jmeter
安装卸载	比较麻烦，文件体积大	比较简便，文件体积小
脚本录制	支持	支持
参数化	支持	支持
集合点	支持	支持
检查点	支持	支持
关联	支持	支持
多协议	支持	支持
IP欺骗	支持	不支持
多线程	支持	支持
报告生成与导出	支持	支持
测试成本	商业软件，成本高	开源软件，测试成本低

## 二、Jmeter 安装配置

1.安装配置好 Jdk

2..Jmeter 下载地址: <http://mirror.bit.edu.cn/apache/jmeter/binaries/>(视频下方获取素材也可以下载)

Java version

JMeter 2.13 requires Java 6 or later

Name	Last modified	Size	Description
 <a href="#">Parent Directory</a>		-	
 <a href="#">HEADER.html</a>	14-Mar-2015 03:31	1.1K	
 <a href="#">apache-jmeter-2.13.tgz</a>	14-Mar-2015 03:30	34M	
 <a href="#">apache-jmeter-2.13.zip</a>	14-Mar-2015 03:30	36M	

Apache/2.2.22 (Ubuntu) Server at mirror.bit.edu.cn Port 81

下载后, 解压文件到任意目录, 避免在一个有空格的路径安装 JMeter, 这将导致远程测试出现问题。●

3..Jmeter 启动: 解压, bin 目录下运行 ApacheJMeter.jar 进行启动。

4.Jmeter 文件目录介绍

◆ bin: 可执行文件目录

#### Bin 目录文件

- jmeter.bat: windows 的启动文件
- jmeter.log: 日志文件
- jmeter.sh: linux 的启动文件
- jmeter.properties: 系统配置文件
- jmeter-server.bat: windows 分布式测试要用到的服务器配置
- jmeter-serve: linux 分布式测试要用到的服务器配置

- ◆ docs: 接口文档目录
- ◆ extras: 扩展插件目录
- ◆ lib: 所用到的插件目录, 里面全是 jar 包, JMeter 会自动在 JMETER\_HOME/lib 和 ext 目录下寻找需要的类
- ◆ Licenses jmeter 证书目录
- ◆ printable\_docs 用户使用手册

## 三、Jmeter 功能概要

### 1. Jmeter 工具组成部分:

- ◆ 资源生成器: 用于生成测试过程中服务器、负载机的资源代码。(LR 中的 VuGen)
- ◆ 用户运行器: 通常是一个脚本运行引擎, 根据脚本要求模拟指定的用户行为。(LR 中的 Controller)
- ◆

- ◆ 报表生成器：根据测试中实时地的数据生成报表，提供可视化的数据显示方式。（LR 中的 Analysis)
- ◆ 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。（LR 中 Load Generators)

**Test Plan (测试计划)**：用来描述一个性能测试，包含与本次性能测试所有相关的功能。也就说本的性能测试的所有内容是于基于一个计划的。（相当于 lr 的一个测试场景）

## 2.Threads (Users)线程 用户

### A. 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。类似 LR 的 init( )

### 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。类似于 LR 中的 end( )

### 3) thread group(线程组).

这个就是我们通常添加运行的线程。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。类似 LR 的 action() ●

### 3.测试片段（Test Fragment）

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。

以下是线程组的 8 类**可执行元件**

### 4.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。如 CSV Data Set config 可以将本地数据文件形成数据池（Data Pool）。

### 5.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 的手段。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。

### 6.前置处理器（Pre 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 URL 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。

### 7.后置处理器（Post Processors）

用于对 Sampler 发出请求后得到的**服务器响应**进行处理。一般用来提取响应中的特定数据（类似 LoadRunner 测试工具中的关联概念）。

## 8.断言 (Assertions)

断言用于检查测试中得到的相应数据等是否符合预期，断言一般用来设置检查点，用以保证性能测试过程中的数据交互是否与预期一致。

## 9.监听器 (Listener)

是用来对测试结果数据进行处理和可视化展示的一系列元件。图行结果、查看结果树、聚合报告。都是我们经常用到的元件。注意：这个监听器可不是用来监听系统资源的元件。

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。

## 10.取样器 (sample)

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。

## 11.逻辑控制器

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。

## 四、Jmeter 脚本录制

### 1.Jmeter 脚本录制

Http 请求+查看结果树

代理服务器操作步骤

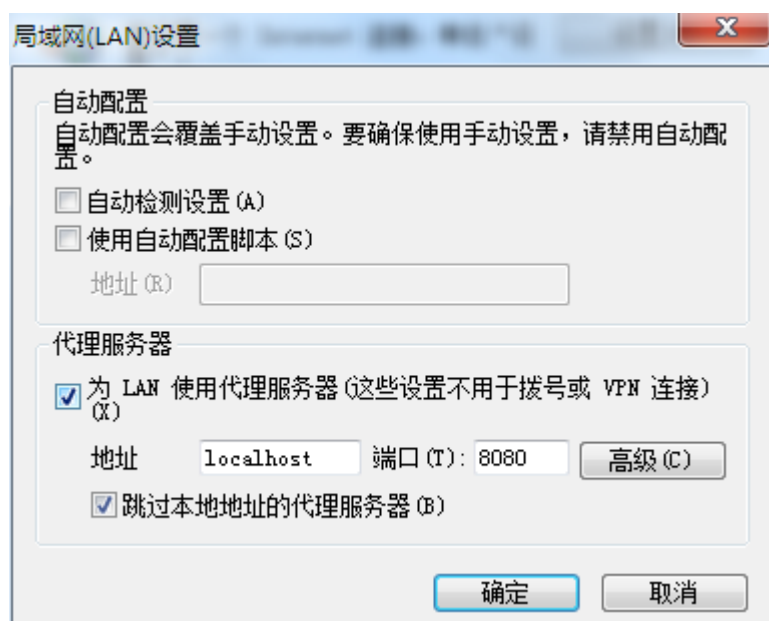
- ◆ 创建一个线程组（右键点击“测试计划” ---> “添加” ----> “线程组”）
- ◆ 创建一个 http 代理服务器（右键点击“工作台” ---> “添加” ---> “非测试元件” ---> “http 代理服务器”）

Tip HTTP 服务器代理设置——分组详解

- 不对样本分组：所有请求全部罗列
- 在组间添加分隔：加入一个虚拟的以分割线命名的动作。
- 每个组放入一个新的控制器：执行时按控制器给输出结果
- 只存储每个组的第一个样本：保存对于一次 url 请求。

设置完后要启动代理服务器，录制完成后记得关闭，

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



- ◆ 在浏览器里对指定的页面进行访问。录制完成后，把浏览器的代理服务器勾去掉。

## 2.Badboy 脚本录制

下载地址 <http://www.badboy.com.au> （或者点击视频左下方【获取素材】按钮来下载）

Badboy 是一个强大的工具，旨在帮助测试和开发复杂的动态应用。Badboy 包括一个简单而全面的捕获/回放界面，强大的负载测试的支持，详细的报告图表等等，从而使 Web 测试和开发变得更加容易。

关于录制时的脚本错误提示



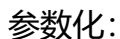
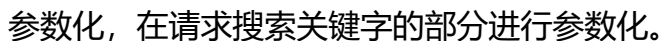


访问者所使用的浏览器不能完全支持页面里的脚本，形成“脚本错误”。遇到“脚本错误”时一般会弹出一个非常难看的脚本运行错误警告窗口，而事实上，脚本错误并不会影响网站浏览，因此这一警告可谓多此一举。要关闭警告则可以在浏览器的工具菜单选择 Internet 选项，然后单击高级属性页。进入到浏览标签，并选中“禁止脚本调试”复选框，以后你就不会再收到这些警告了。

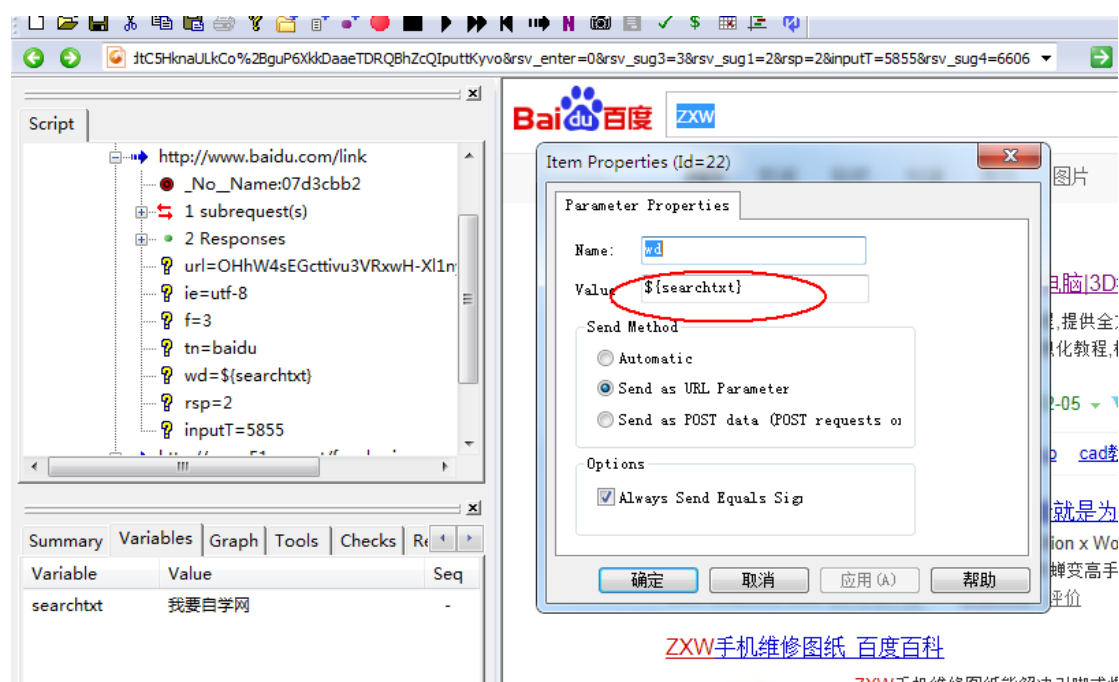
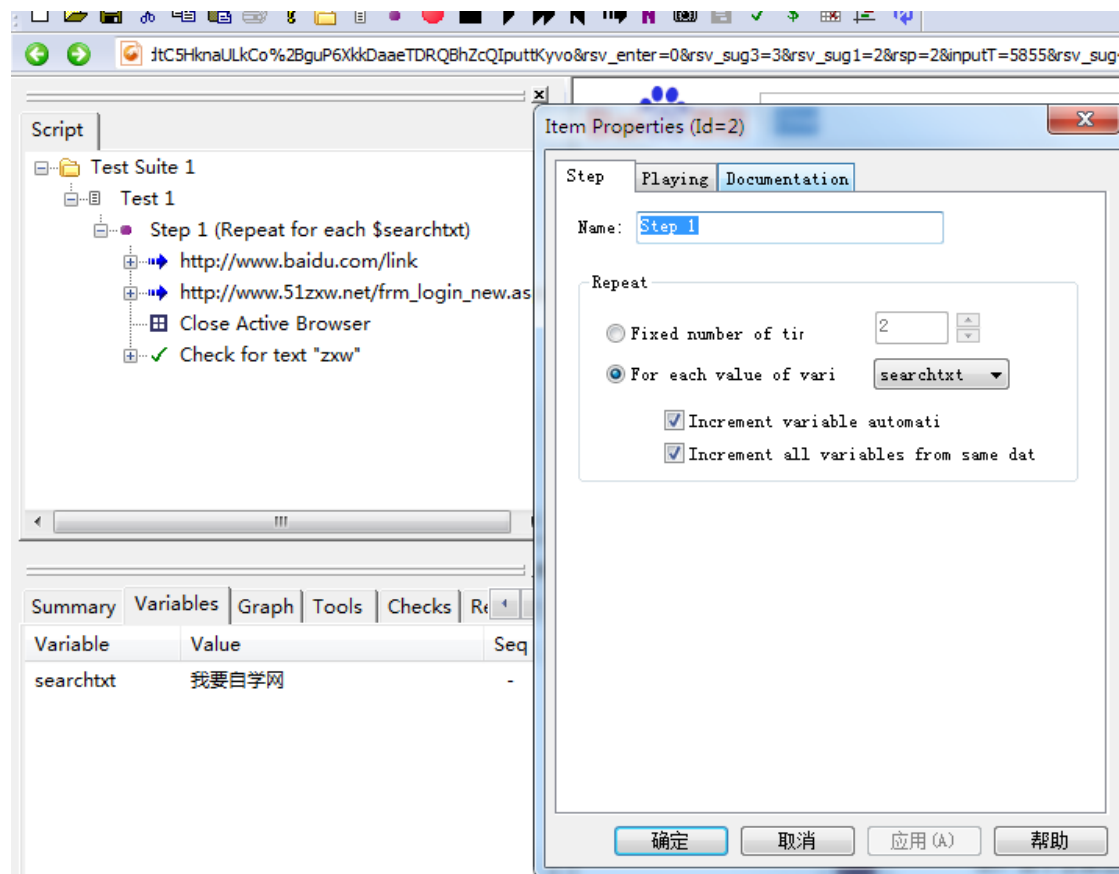
脚本录制完成后，导出为 Export to Jmeter （见脚本 haosou.bx）

## 五、badboy 检查点与参数化

检查点设置：选择要检查的文字，然后在 Tools>step1 里添加断言,再回放



找到请求的字段，然后对其值进行设定。参数化替换\${参数名称} 最后在 step 1 属性里进行设置回放次数和参数化的变量名称。最后检查点也要进行参数化,以及整个 step 属性的设置



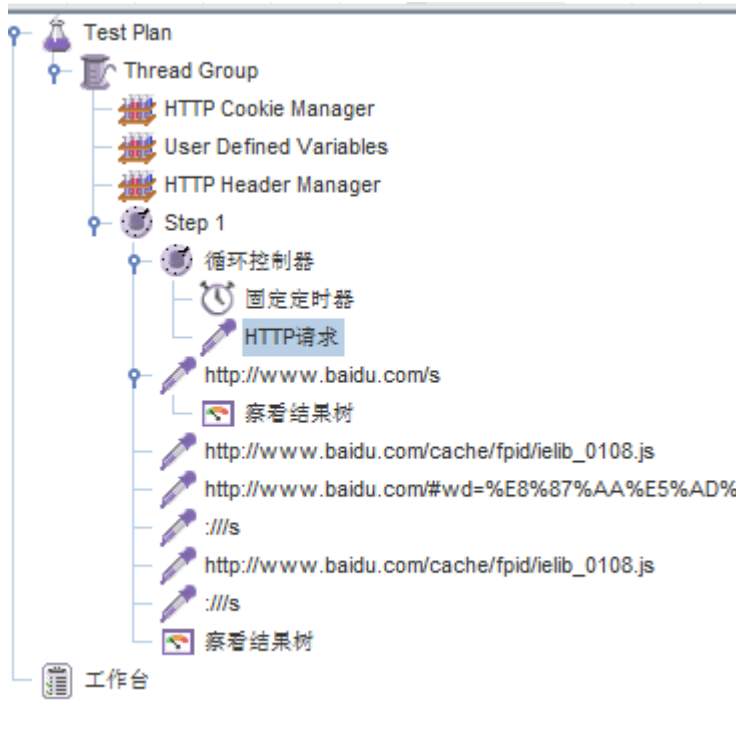
## 六、Jmeter 元件作用域和执行顺序

### 1. 元件作用域

8 类可被执行的元件 (测试计划与线程组不属于**可执行元件**)，这些元件中，取样器 (sampler) 是典型的不与其它元件发生交互作用的元件，逻辑控制器只对其子节点的取样器有效，而其它元件 (配置元件、定时器、断言、监听器、) 需要与取样器 (sampler) 等元件交互。

在 jmeter 中，元件的作用域是靠测试计划的树型结构中元件的**父子关系**来确定的，作用域的原则是：

- 取样器 (sampler) 元件不和其它元件相互作用，因此**不存在作用域**的问题。
- 逻辑控制器 (Logic Controller) 元件只对**其子节点**中的**取样器和逻辑控制器**作用。
- 除取样器和逻辑控制器元件外，其他 6 类元件，如果是某个取样器的子节点，则该元件对其父子节点起作用。如果其父节点不是取样器，则其作用域是该元件父节点下的其他所有后代节点 (包括子节点，子节点的子节点等)。



最后一个查看结果树的作用域为循环控制器下面所有取样器的结果

## 2.元件执行顺序

- (1) 配置元件 (config elements )
- (2) 前置处理程序 (Per-processors)
- (3) 定时器 (timers )
- (4) 取样器 (Sampler)
- (5) 后置处理程序 (Post-processors)
- (6) 断言 (Assertions)
- (7) 监听器 (Listeners)

关于执行顺序，有两点需要注意：

前置处理器、后置处理器和断言等元件功能对**取样器**作用，因此，如果在它们的作用域内没有任何取样器，则不会被执行。

如果在同一作用域范围内有多个**同一类型**的元件，则这些元件按照它们在测试计划中的上下顺序依次执行。

## 七、Jmeter 性能测试基础实战

**1.测试需求：**测试 20 个用户访问 <http://www.51zxw.net> 在负载达到 30 QPS 时的平均响应时间。

**QPS** : Query Per Second 每秒查询率。是一台查询服务器每秒能够处理的查询次数。在因特网上，作为域名系统服务器的性能经常用每秒查询率来衡量。

### 2.测试步骤：

#### 第一步：添加线程组

线程组主要包含三个参数：线程数、准备时长 (Ramp-Up Period(in seconds)) 、循环次数。

- ❖ 线程数：虚拟用户数。一个虚拟用户占用一个进程或线程。设置多少虚拟用户数在这里也就是设置多少个线程数。
- ❖ 准备时长（单位为 s）：设置的虚拟用户数需要多长时间全部启动。如果线程数为 20 ，准备时长为 10 ，那么需要 10 秒钟启动 20 个线程。也就是每秒钟启动 2 个线程。

- ❖ 循环次数：每个线程发送请求的次数。如果线程数为 20，循环次数为 5，那么每个线程发送 5 次请求。总请求数为  $20 \times 5 = 100$ 。如果勾选了“永远”，那么所有线程会一直发送请求，一到选择停止运行脚本。

## 第二步：增添 HTTP 请求



一个 HTTP 请求有着许多的配置参数，下面将详细介绍：

- ✧ **名称**：本属性用于标识一个取样器，建议使用一个有意义的名称。
- ✧ **注释**：对于测试没有任何作用，仅用户记录用户可读的注释信息。
- ✧ **服务器名称或 IP**：HTTP 请求发送的目标服务器名称或 IP 地址。
- ✧ **端口号**：目标服务器的端口号，默认值为 80。
- ✧ **Timeouts (milliseconds)**：设置请求和响应的超时时间
- ✧ **协议**：向目标服务器发送 HTTP 请求时的协议，可以是 http 或者是 https，默认值为 http。

- ✧ **方法**: 发送 HTTP 请求的方法, 可用方法包括 GET、POST、HEAD、PUT、OPTIONS、TRACE、DELETE 等。
- ✧ **Content encoding** : 内容的编码方式, 默认值为 iso8859
- ✧ **路径**: 目标 URL 路径 (不包括服务器地址和端口)
- ✧ **自动重定向**: 如果选中该选项, 当发送 HTTP 请求后得到的响应是 302/301 时, JMeter 自动重定向到新的页面。
- ✧ **Use keep Alive** : 当该选项被选中时, jmeter 和目标服务器之间使用 Keep-Alive 方式(又称持久连接、连接重用)进行 HTTP 通信, 默认选中。
- ✧ **Use multipart/form-data for HTTP POST** : 当发送 HTTP POST 请求时, 使用 Use multipart/form-data 方法发送, 默认不选中。
- ✧ **同请求一起发送参数** : 在请求中发送 URL 参数, 对于带参数的 URL , jmeter 提供了一个简单的对参数化的方法。用户可以将 URL 中所有参数设置在本表中, 表中的每一行是一个参数值对 (对应 URL 中的 名称=值) 。
- ✧ **同请求一起发送文件**: 在请求中发送文件, 通常 HTTP 文件上传行为可以通过这种方式模拟。
- ✧ **从 HTML 文件获取所有有内含的资源**: 当该选项被选中时, jmeter 在发出 HTTP 请求并获得响应的 HTML 文件内容后, 还对该 HTML 进行分析并获取 HTML 中包含的所有资源 (图片、flash 等) , 默认不选中, 如果用户只希望获取页面中的特定资源, 可以在下方的 Embedded URLs must match 文本框中填入需要下载的特定资源表达式, 这样, 只有能匹配指定正则表达式的 URL 指向资源会被下载。
- ✧ **用作监视器**: 此取样器被当成监视器, 在 Monitor Results Listener 中可以直接看到基于该取样器的图形化统计信息。默认为不选中。

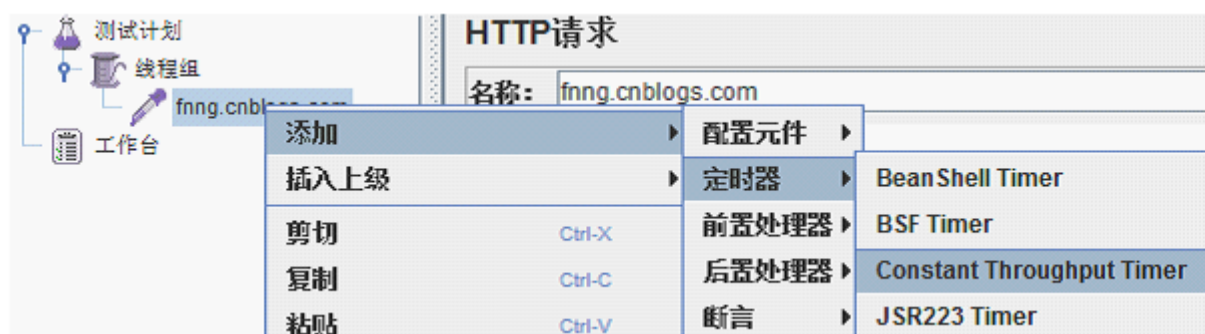


- ✧ **Save response as MD5 hash?** : 选中该项, 在执行时仅记录服务端响应数据的 MD5 值, 而不记录完整的响应数据。在需要进行数据量非常大的测试时, 建议选中该项以减少取样器记录响应数据的开销。

tips 默认时间单位是毫秒 报告输出文件后缀 .jtl

### 第三步: 设置 QPS 限制

Jmeter 提供了一个非常有用的定时器, 称为 Constant Throughput Timer (常数吞吐量定时器), 该定时器可以方便地控制给定的取样器发送请求的吞吐量。



Constant Throughput Timer 的主要属性介绍:

Target throughput (in samples per minute) : 目标吞吐量。注意这里是每分钟发送的请求数, 实际填的数值为:  $60 \times \text{QPS}$  其次 Calculate Throughput based on : 有 5 个选项, 分别是:

- **This thread only** : 控制每个线程的吞吐量, 选择这种模式时, 总的吞吐量为设置的 target Throughput 乘以该线程的数量。
- **All active threads** : 设置的 target Throughput 将分配在每个活跃线程上, 每个活跃线程在**上一次运行结束后**等待合理的时间后再次运行。活跃线程指同一时刻同时运行的线程。

- **All active threads (shared)** : 与 All active threads 的选项基本相同, 唯一的区别是, 每个活跃线程都会在**所有活跃线程**上一次运行结束后等待合理的时间后再次运行。
- **All active threads in current thread group** : 设置的 target Throughput 将分配在当前线程组的每一个活跃线程上, 当测试计划中只有一个线程组时, 该选项和 All active threads 选项的效果完全相同。
- **All cative threads in current thread group (shared)** : 与 All active threads in current thread group 基本相同, 唯一的区别是, 每个活跃线程都会在**所有活跃线程**的上一次运行结束后等待合理的时间后再次运行。

#### 第四步: 添加监视器

脚本的主要部分设置完成后, 需要通过某种方式获得性能测试中的测试结果, 在本例中, 我们关心的是请求的响应时间。

Jmeter 中使用监听器元件收集取样器记录的数据并以可视化的方式来呈现。Jmeter 有各种不同的监听器类型, 因为上 HTTP 请求, 我们可在添加聚合报告, 更为直观的查看测试结果。

添加聚合报告, 右键点击线程组, 在弹的菜单 (添加--->监听器--->聚合报告) 中选择聚合报告。

添加查看结果树 (添加--->监听器--->查看结果树)

#### 第五步: 运行脚本

#### 第六步: 聚合报告分析



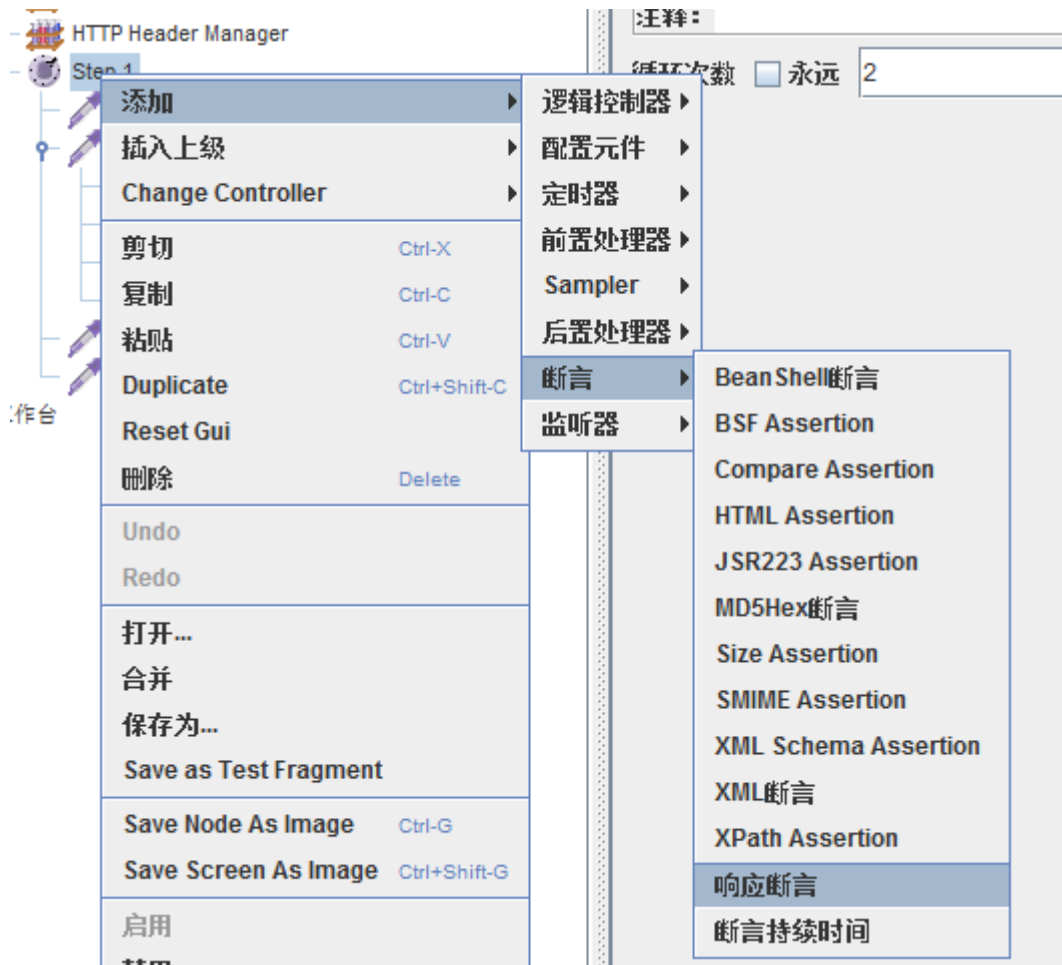
Term	Definition
Label	每个 JMeter 的 element(例如 HTTP Request) 都有一个 Name 属性, 这里显示的就是 Name 属性的值
#Samples	表示你这次测试中一共发出了多少个请求, 如果模拟 10 个用户, 每个用户迭代 10 次, 那么这里显示 100
Average	平均响应时间——默认情况下是单个 Request 的平均响应时间, 当使用了 Transaction Controller 时, 也可以以 Transaction 为单位显示平均响应时间。
Median	中位数, 也就是 50% 用户的响应时间。
90%Line	90% 用户的响应时间。
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), 当使用了 Transaction Controller 时, 也可以表示类似 LoadRunner 的 Transaction per Second 数
KB/sec	每秒从服务器端接收到的数据量, 相当于 LoadRunner 中的 Throughput/Sec

响应时间单位: 毫秒 见脚本: 访问我要自学网\_30QPS.jmx

## 八 Jmeter 断言 (检查点)

断言是在请求的返回层面增加一层判断机制。因为请求成功了, 并不代表结果一定正确, 因此需要检测机制提高测试准确性。下面介绍常用的 jmeter 三种断言

## 1.响应断言：



### 模式匹配规则

- ◆ 包括：返回结果包括你指定的内容
- ◆ 匹配：根据指定内容进行匹配
- ◆ Equals：返回结果与你指定结果一致
- ◆ Substring：返回结果是指定结果的字符串
- ◆ 否：不进行匹配

## 2.Size Assertion (Size 断言)

### Size Assertion

名称:

注释:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable

Response Size Field to Test

☒ Full Response ☐ Response Headers ☐ Response Body ☐ 响应代码 ☐ 响应信息

Size to Assert

字节大小:

比较类型

☒ =  
☐ !=  
☐ >  
☐ <  
☐ >=  
☐ <=

## 3.Duration Assertion (持续时间断言)

### 断言持续时间

名称:

注释:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only

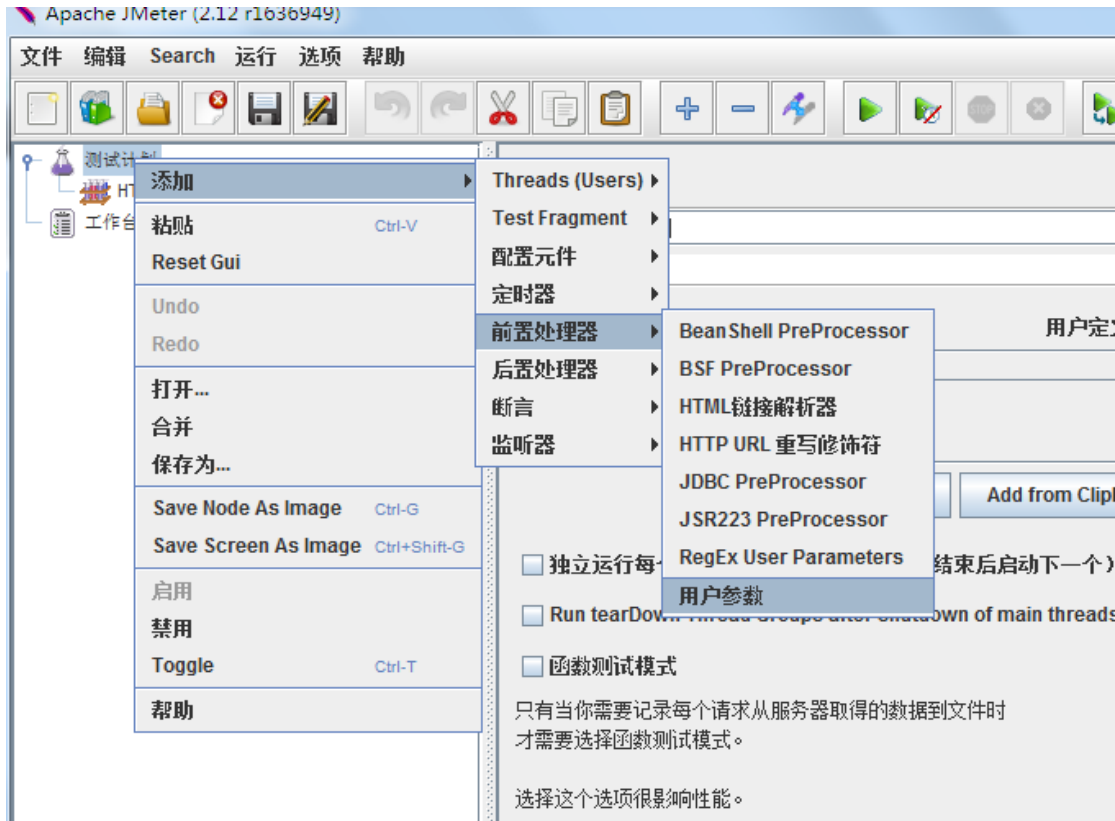
断言持续时间

持续时间 (毫秒):

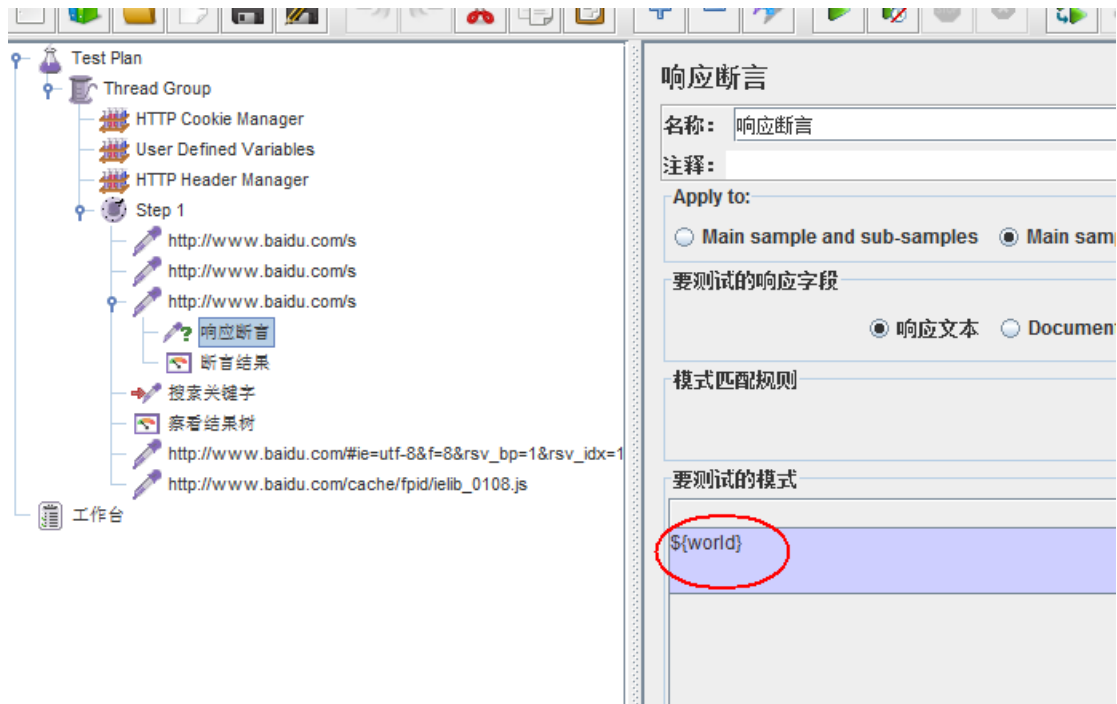
如果响应时间大于设置的响应时间，则断言失败，否则成功！（见代码 Assertion.jmx）

## 九 Jmeter 参数化

### 1. 用户参数

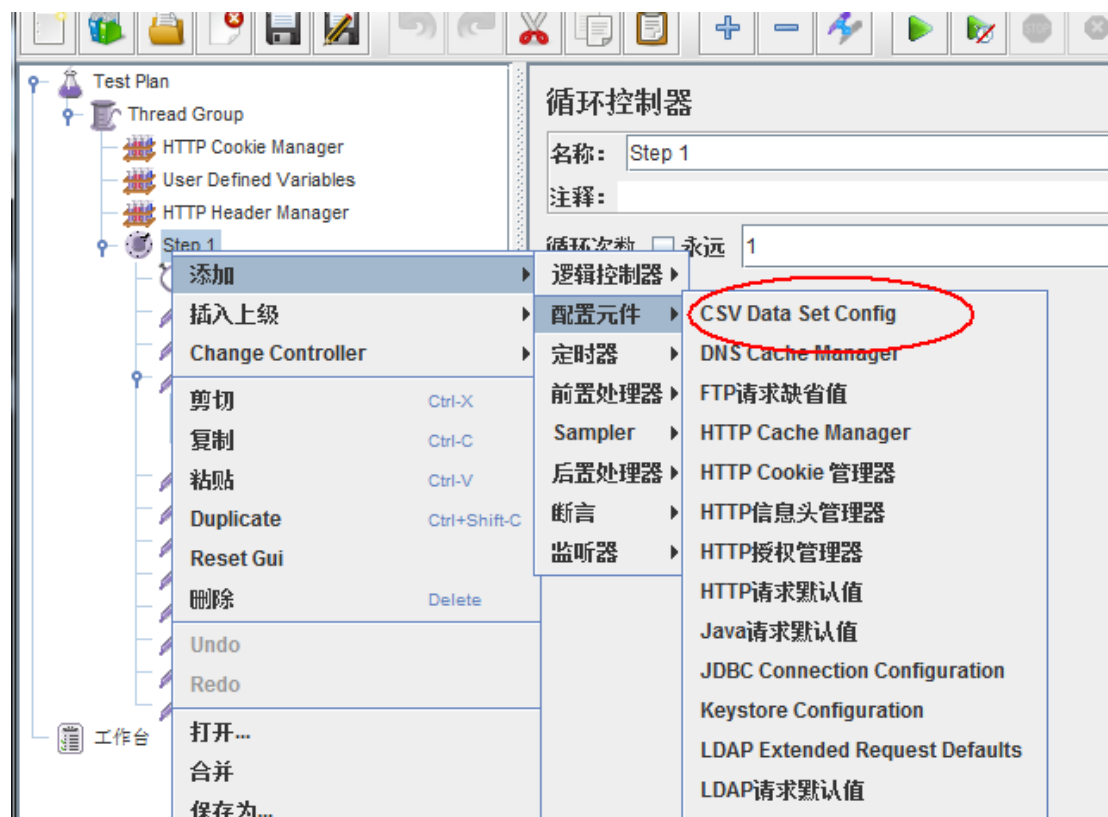


响应断言，断言也要记得变量替换



首先 前置处理器-用户参数

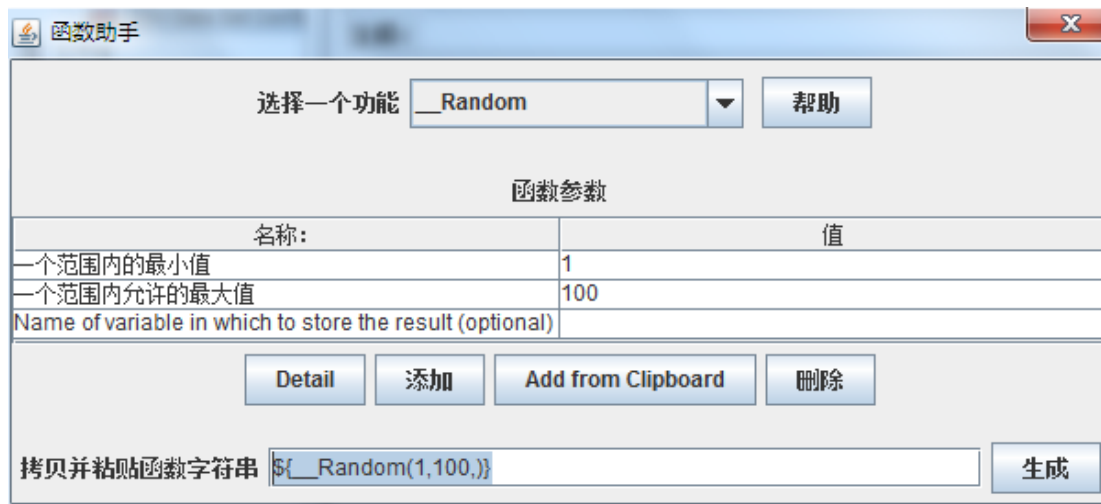
## 2.CSV 数据配置



- ❖ Filename --- 参数项文件
- ❖ File Encoding --- 文件的编译方法，一般为空
- ❖ Variable Names --- 文件中各列所表示的参数项；各参数项之间利用逗号分隔；参数项的名称应该与 HTTP Request 中的参数项一致。
- ❖ Delimiter --- 如文件中使用的是逗号分隔，则填写逗号；如使用的是 TAB，则填写\t；
- ❖ Recycle on EOF? --- True=当读取文件到结尾时，再重头读取文件,False=当读取文件到结尾时，停止读取文件
- ❖ Stop thread on EOF? --- 当 Recycle on EOF?一项为 False 时起效；True=当读取文件到结尾时，停止进程。



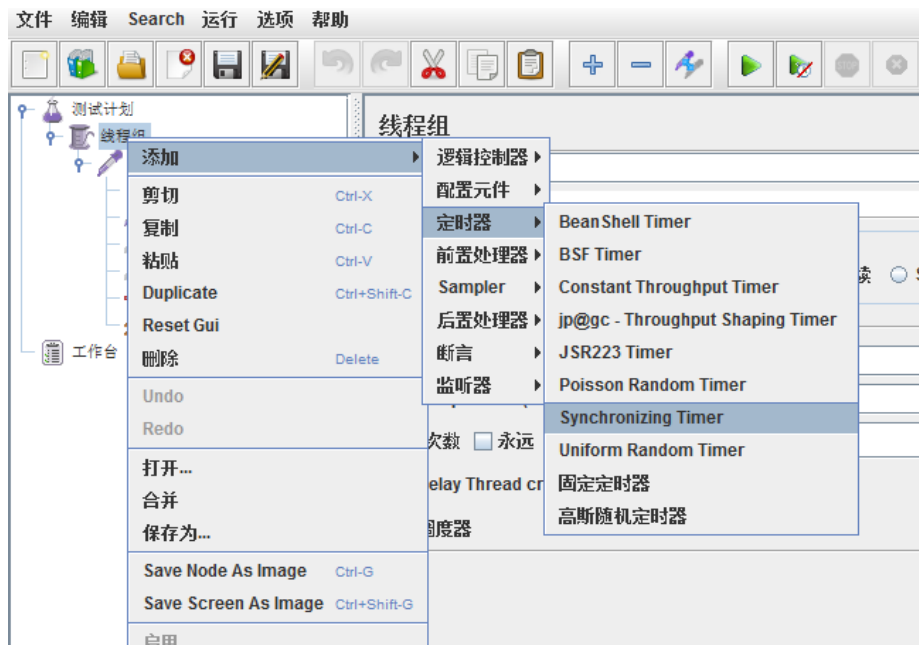
### 3.随机参数化



详见脚本 Parameter.jmx

## Jmeter 集合点

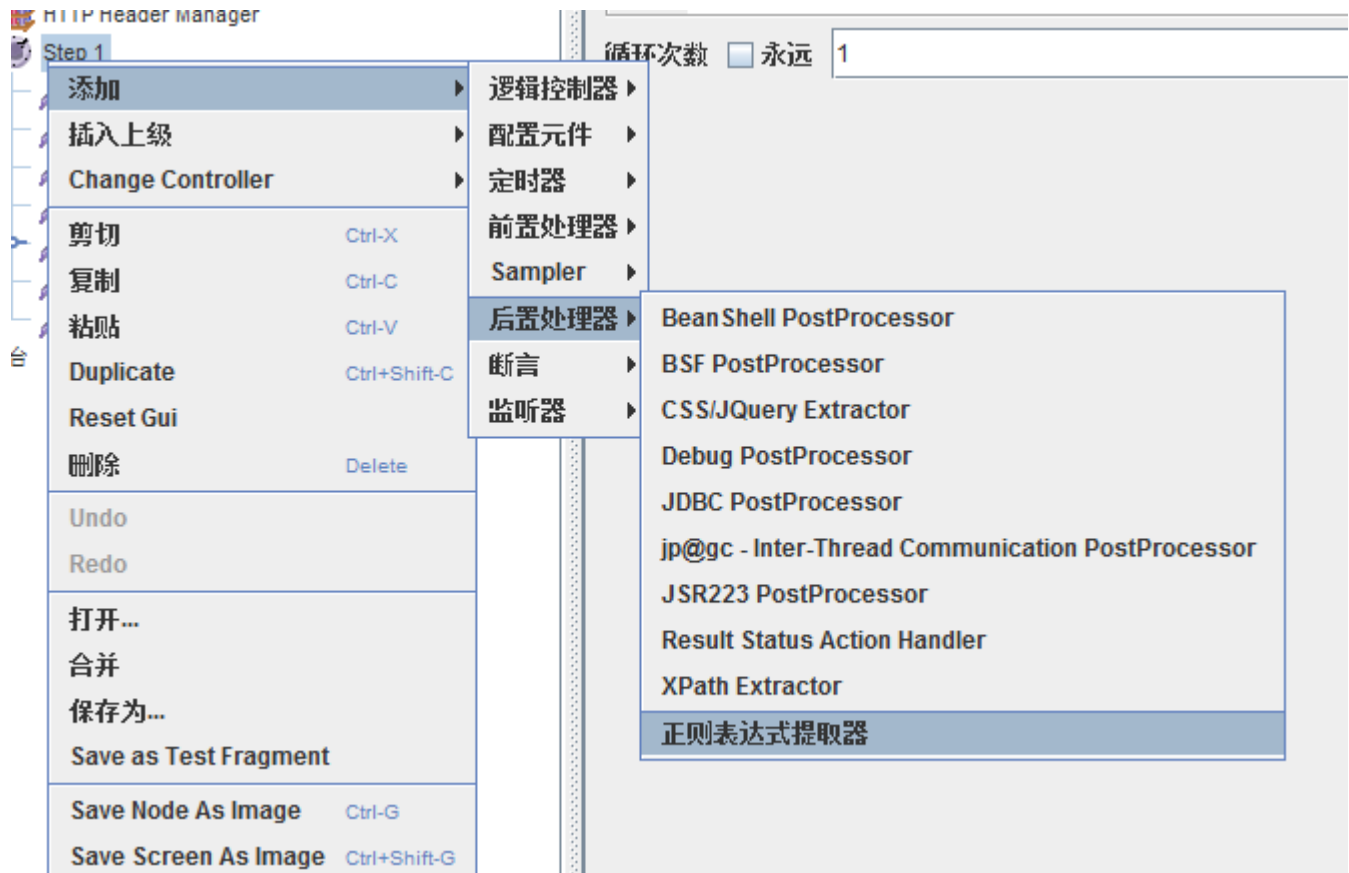
操作步骤: step1---->定时器---->Synchronizing Timer



注意: 集合点要放在需要集合的元件前面。

## Jmeter 关联

### 1.正则表达式提取器



#### 正则表达式提取器

名称:	正则表达式提取器
注释:	
Apply to:	<input type="radio"/> Main sample and sub-samples <input checked="" type="radio"/> Main sample only <input type="radio"/> Sub-samples only <input type="radio"/> JMeter Variable
要检查的响应字段	<input checked="" type="radio"/> 主体 <input type="radio"/> Body (unescaped) <input type="radio"/> Body as a Document <input type="radio"/> 信息头 <input type="radio"/> Request Headers <input type="radio"/> URL <input type="radio"/> 响应代码 <input type="radio"/> 响应信息
引用名称:	
正则表达式:	
模板:	
匹配数字 (0代表随机):	
缺省值:	

#### 正则表达式部分配置说明

- ◆ (1) 引用名称: 下一个请求要引用的参数名称, 如填写 activityID, 则可用\${activityID}引用它。

◆ (2) 正则表达式:

()括起来的部分就是要提取的。

.匹配任何字符串。

+：一次或多次。

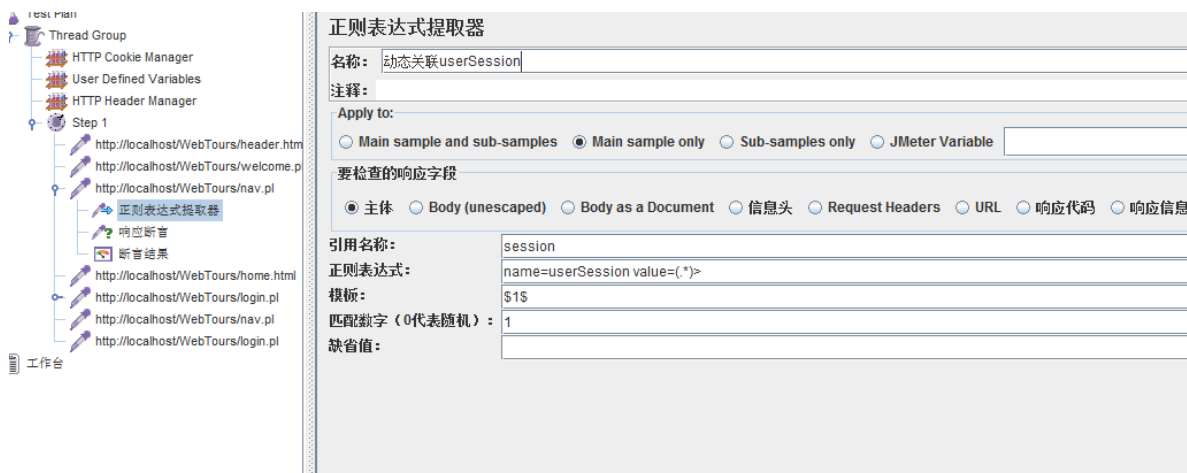
?：在找到第一个匹配项后停止。

◆ (3) 模板：用\$\$引用起来，如果在正则表达式中有多个正则表达式（多个括号括起来的东东），则可以是\$2\$3\$等等，表示解析到的第几个值给 title。如：\$1\$表示解析到的第 1 个值

◆ (4) 匹配数字：0 代表随机取值，1 代表全部取值，

◆ (5) 缺省值：如果参数没有取得到值，那默认给一个值让它取。

Webtours 关联要在 in =home 来进行设置，



关联设置如上所示 正则表达式 name=userSession value=(.\*)>

## 十、Jmeter 图形监控扩展

插件下载 <http://jmeter-plugins.org/downloads/all/>

## Latest Stable Release

	<a href="#">JMeterPlugins-Standard-1.2.1.zip</a> , 1.15 MB, Mar 09, 2015, <i>Standard Set</i>
	<a href="#">ServerAgent-2.2.1.zip</a> , 3.51 MB, Jul 12, 2013, <i>PerfMon Agent to use with Standard Set</i>
	<a href="#">JMeterPlugins-Extras-1.2.1.zip</a> , 1.31 MB, Mar 09, 2015, <i>Extras Set</i>
	<a href="#">JMeterPlugins-ExtrasLibs-1.2.1.zip</a> , 4.78 MB, Mar 09, 2015, <i>Extras with Libs Set</i>
	<a href="#">JMeterPlugins-WebDriver-1.2.1.zip</a> , 10.38 MB, Mar 09, 2015, <i>WebDriver Set</i>
	<a href="#">JMeterPlugins-Hadoop-1.2.1.zip</a> , 10.86 MB, Mar 09, 2015, <i>Hadoop Set</i>

- (1) 首先将 JmeterPlugging.jar 包复制到 Jmeter 的 Lib 目录下面的 ext 目录下面, 然后重新启动
- (2) Jmeter 将 serverAgent 目录及下面的文件复制到我们测试的服务器上, 然后点击打开 serverAgent.bat, 它默认端口 4444 (特别是对于 PerfMon Metrics Collector 测试一定要启动 serverAgent, )



The screenshot shows the JMeter GUI with the 'Add' menu open. The 'Listeners' category is selected, and a list of listeners is displayed on the right. The 'jp@gc - Transactions per Second' listener is highlighted.

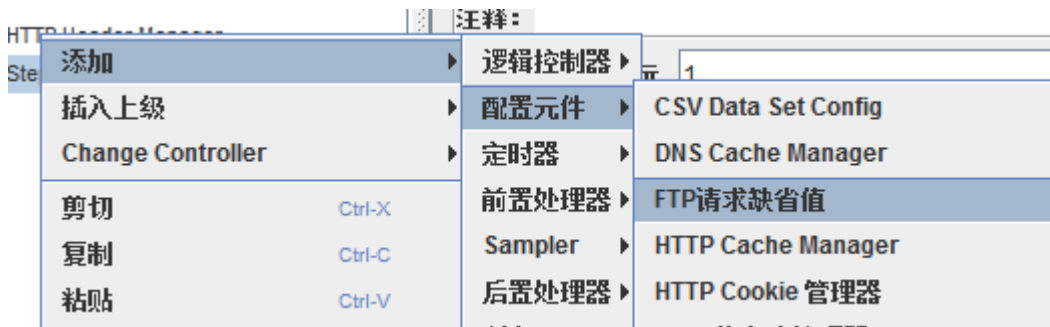
Listener Name	Shortcut
jp@gc - Console Status Logger	
jp@gc - Flexible File Writer	
jp@gc - Hits per Second	
jp@gc - Loadosophia.org Uploader	
jp@gc - PerfMon Metrics Collector	
jp@gc - Response Times Over Time	
jp@gc - Synthesis Report (filtered)	
<b>jp@gc - Transactions per Second</b>	
JSR223 Listener	
Response Time Graph	
Simple Data Writer	
Spline Visualizer	
Summary Report	
保存响应到文件	
图形结果	
察看结果树	

(3) 添加相应的监控图表，运行脚本。

## 十一、Jmeter FTP 服务器连接

1.创建一个线程组

2.线程组--->添加--->配置元件--->FTP 请求缺省值。



3.线程组--->添加--->Sampler--->FTP 请求



### FTP请求

名称:	FTP请求		
注释:			
服务器名称或IP:		端口号:	
Remote File:			
Local File:			
Local File Contents:			
<input checked="" type="radio"/> get(RETR) <input type="radio"/> put(STOR) <input type="checkbox"/> Use Binary mode ? <input type="checkbox"/> Save File in Response ?			
登陆配置			
用户名			
密码			

说明：

- IP :为你 FTP 服务的 IP
- Remote file: 为你 FTP 服务器上的一个文件。
- local file:为本地你存放到本机上的路径。
- 选择 get(RETR) 为下载方式。
- 登录配置：填写你的 FTP 服务器的用户名密码。

4.按照第 3 步再添加一个“FTP”请求。选择 put 为上传方式。

5.添加一个监控器：线程组--->添加--->监控器--->Spline Visualizer

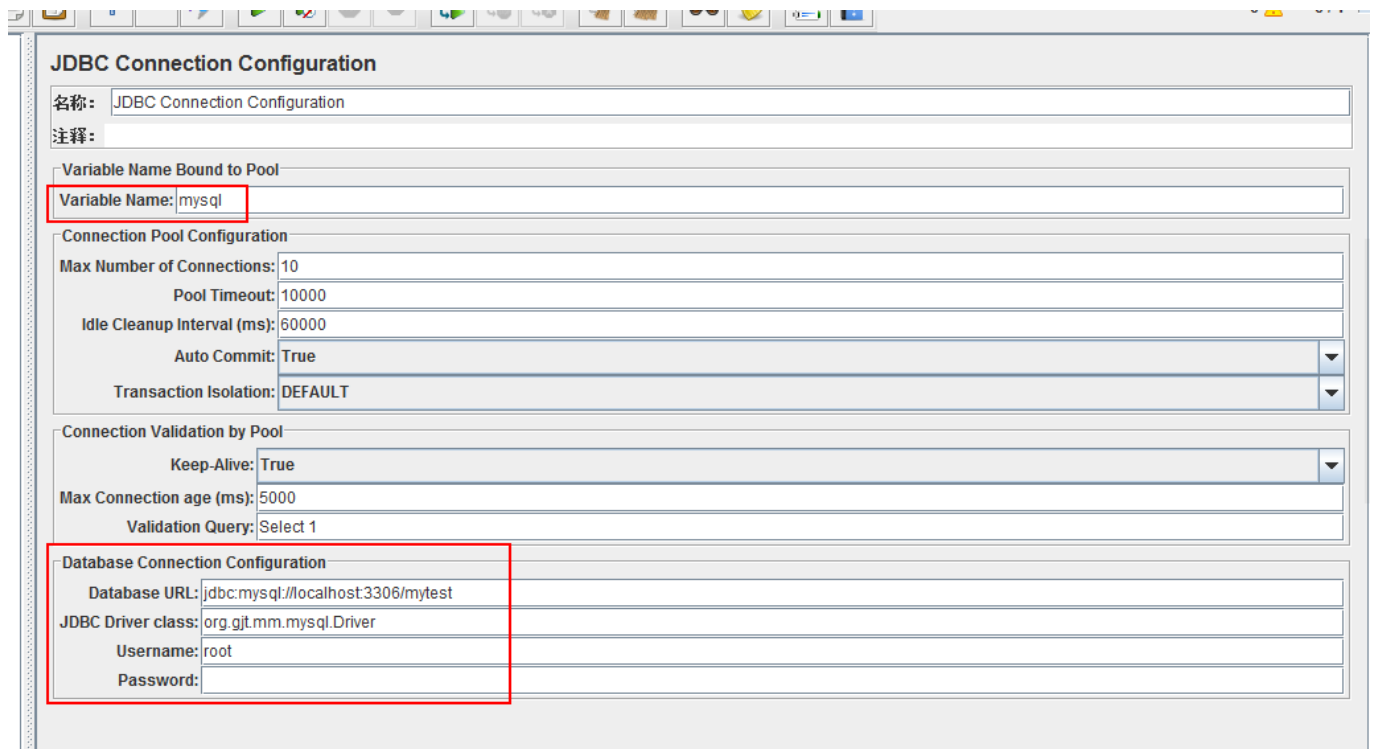


## 十二、Jmeter mysql 测试

- 1.准备一个有测试数据表的 mysql 数据库。
- 2.在测试计划面板点击“浏览...”按钮，将你的 JDBC 驱动添加进来。

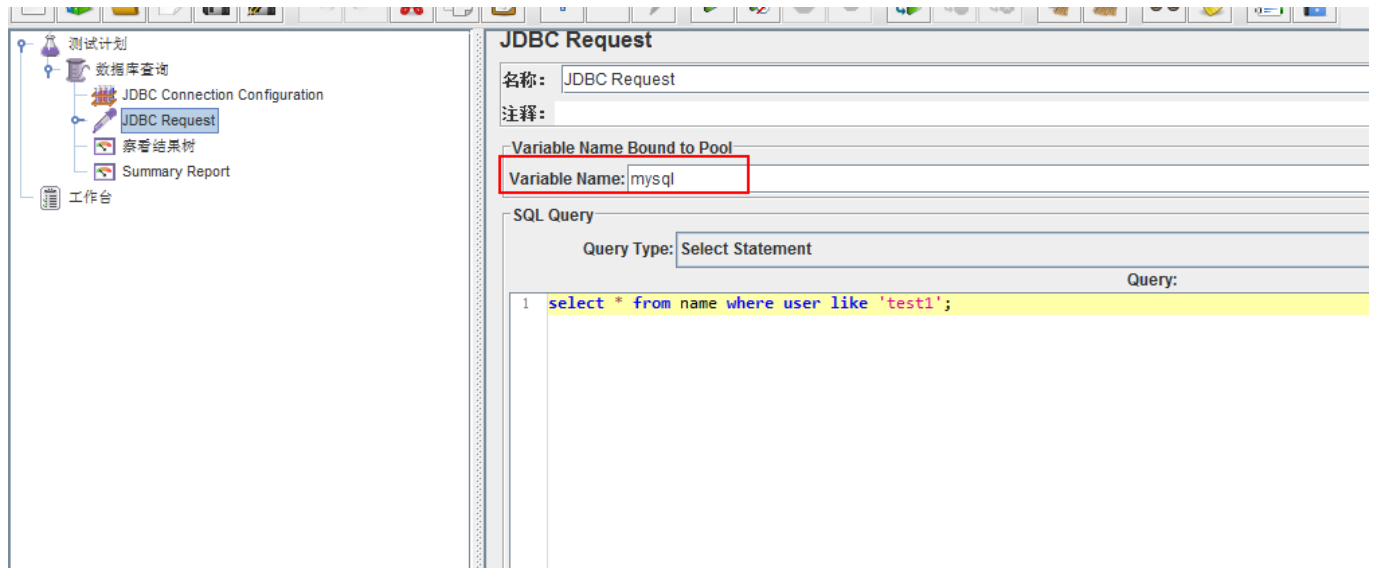


3.添加一个线程组, 右键点击“线程组”, 在下面添加一个“JDBC Connection Configuration”并配置好各项参数。



4.右键点击“线程组”, 在下面添加一个“JDBC request” 输入查询语句

如果需要通过实现同时多个不同用户使用不同的 SQL, 可以通过把整条 SQL 语句参数化来实现, 例如把 JDBC Request 的 Query 使用参数化代替\${SQL\_Statement}



6.右键点击线程组---->添加--->断言---->响应断言。

7.添加一些监听器

- ◆ 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。

见脚本：mysql\_test.jmx

## 十三、Jmeter 分布式测试

### 背景

由于 Jmeter 本身的瓶颈，当需要模拟数以千计的并发用户时，使用单台机器模拟所有的并发用户就有些力不从心，甚至还会引起 JAVA 内存溢出的错误。要解决这个问题，可以使用分布式测试，运行多台机器运行所谓的 Agent 来分担 JMeter 自身的压力，并借此来获取更大的并发用户



数，但是需要进行相关的一些修改，具体如下。

## 操作步骤

1.安装 JMeter, 并确定其中一台机器作为 Controller, 其他的机器作为 Agent。然后运行所有 Agent 机器上的 JMeter-server.bat 文件——假定我们使用两台机器 192.168.0.11 和 192.168.0.12 作为 Agent; (Agent 机器上必须安装 jdk,并设置环境变量)

2.在 Controller 机器的 %JMeter\_home%/bin 下, 编辑 JMeter.properties 中 “remote\_hosts=127.0.0.1” 。其中的 127.0.0.1 表示运行 JMeter Agent 的机器, 这里需要修改为 “remote\_hosts=192.168.0.11:1099,192.168.0.12:1099” ——其中的 1099 为端口号。

3、启动 controller 机器上的 jmeter.bat, 选择菜单 Run 中 “Remote Start” 中的 192.168.0.11:10099 和 192.168.0.12: 1099 来运行 Agent。

5.如果要想让某个电脑执行, 可以点击改电脑的 IP 地址就可以, 如果两个都要执行, 可以点击

Run 菜单下的 “远程运行全部”  菜单

6、有时候用作代理的机器太少, 仍不能满足需要, 则需要将作为 Controller 的电脑也当作 Agent, 则同样需要修改 JMeter.properties 文件, 将 Controller 的 IP 地址写入。同时, 这个时候, 需要先打开 Controller 电脑中 JMeter 下 bin 目录下的 jmeter-server.bat, 然后再打开 JMeter.bat, 此时, 进入 Run -> Remote Start 菜单, 可以看到 Controller 也作为远程机器进行运行。

## 常见问题:

1、确定在 controller 机器上安装 jdk,版本和 jmeter 一致

2、Agent 机器启动 Jmeter\_server.bat 时，后台提示："could not find ApacheJmeter\_core.jar"

解决方法：这个是开始没有找到 ApacheJmeter\_core.jar，如果不希望看到 Could not find 的字样，需要添加环境变量 JMETER\_HOME，路径为 bin 目录的上一级目录，这样启动 jmeter-server 服务时，就不会看到 could not Found ApacheJMeter\_core.jar。

3、Jmeter 分布式控制过程中，各个 Agent 启动的线程数等于线程组中的配置。

## 十四、Jmeter Java 工程测试

1.在 eclipse 里面新建一个工程

2.从 Jmeter 的安装目录 lib/ext 中引入两个文件"ApacheJMeter\_core.jar"和"ApacheJMeter\_java.jar"到 java 工程

3.编写 jmeter 辅助函数

### JMeter Java Sampler 介绍

Arguments	<b>getDefaultParameters()</b> 用于获取界面的参数
SampleResult	<b>runTest(JavaSamplerContext context)</b> 类似于LR的Action，result.sampleStart()一个事务开始，result.sampleEnd()一个事务 结束
void	<b>setupTest(JavaSamplerContext context)</b> 初始化方法，类似于LR的init和JUnit中的setUp()
void	<b>teardownTest(JavaSamplerContext context)</b> 类似于LR的end和JUnit中的tearDown()

执行的先后顺序为：

getDefaultParameters() --> setupTest(JavaSamplerContext context) --> runTest(JavaSamplerContext context) --> teardownTest(JavaSamplerContext context)

常用的方法：

①、addArgument("name", "value") 定义参数

②、sampleStart() 定义事务的开始，类似于 LR 的 lr\_start\_transaction，和 LR 一样事务间不要放无关代码

③、sampleEnd() 定义事务的结束，类似于 LR 的 lr\_end\_transaction

④、setSuccessful(true、false) 设置运行结果的成功或失败，Jmeter 统计成功失败的次数，在聚合报告中能够体现。

4.将工程导出成 Jar，放置于 \$jmeter 安装目录\$/lib/ext/ 下，其它依赖的 Jar 放置于 \$安装目录/lib/ 下

5.运行脚本

- 选中主界面左侧的“测试计划”，右键菜单->添加->Threads(Users)->线程组
- 再选中刚才新增的"线程组"，右键菜单->添加->Sampler->Java 请求
- 再选中刚才新增的"Java 请求"，右键菜单->添加->监视器->聚合报告
- 在"Java 请求"选项卡中可以选择你想测试的类名；在"线程组"选项卡中可以输入想循环的次数及并发线程数
- 一切就绪后，点击菜单栏上的"运行"->启动

## 十五、Jmeter 函数

JMeter 函数是一些能够转化在测试树中取样器或者其他配置元件的域的特殊值。一个函数的调用就像这样：

`$_functionName(var1,var2,var3)`，`__functionName` 匹配函数名，圆括号内设置函数的参数，例如

`$_time(YMD)`实际参数因函数而不同。不需要参数的函数使圆括号内为空，例如`$_theadNum()`。

Jmeter 函数有两种函数：**自定义静态值**（用户变量）和**内置函数**。

自定义静态值允许当一个测试树编译提交运行时，自定义变量被它们的静态值代替。这个替代在测试运行开始时发生一次。内置函数允许写进任何非控制器测试组件的任何域，这包括取样器，定时器，监听器，断言等

注意：如果使用和内置函数同样的名字定义一个自定义变量，你的自定义静态变量会覆盖内置函数，但不建议名字相同。

## Jmeter 常用内置函数

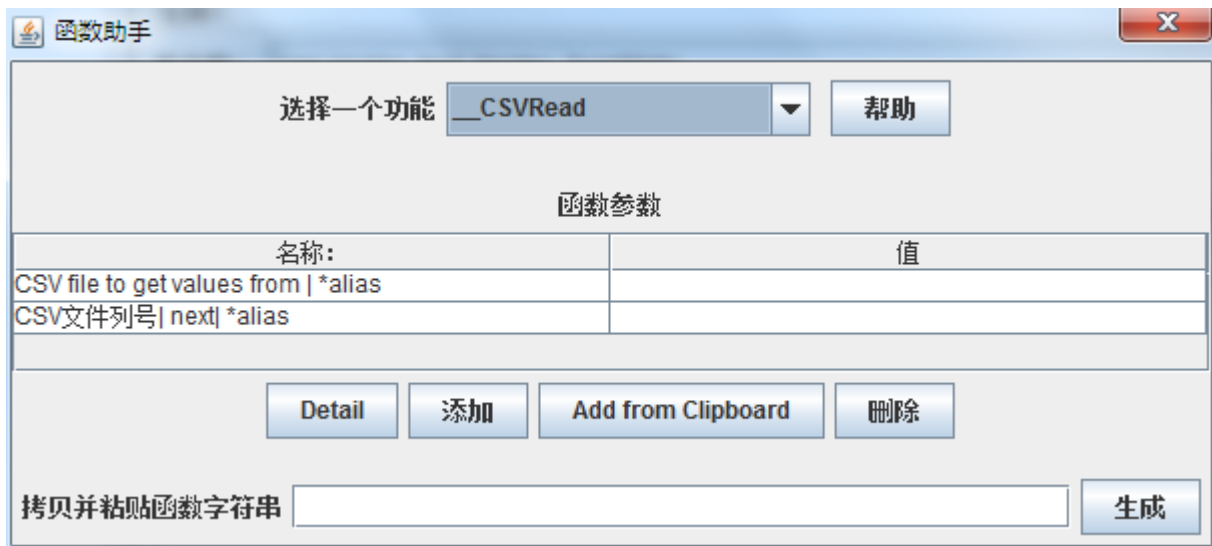
### 1. \_\_csvRead 函数

\_\_cvsRead 函数是从外部读取参数，\_\_csvRead 函数可以从一个文件中读取多个参数。

(1) 先新建一个文件，例如 csvRead.txt，里面的数据存放如下：

11, 22, 33, 44

(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：



说明：

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

(3) 设置好相关参数，将生成的函数复制到相应的请求参数之中。

### 2. \_StringFromFile 函数

StringFromFile 函数是从一个文件中读取一个字符串,用来实现参数化,如果读取或者打开这个文件发生错误时，

将会返回 "\*\*\*ERR\*\* " 字符串

函数助手

选择一个功能 \_StringFromFile 帮助

函数参数

名称:	值
输入文件的全路径	
Name of variable in which to store the result (optional)	
Start file sequence number (opt)	
Final file sequence number (opt)	

Detail 添加 Add from Clipboard 删除

拷贝并粘贴函数字符串  生成

1、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：选择\_StringFromFile

**输入文件的全路径：**输入读取文件的绝对路径+文件名。

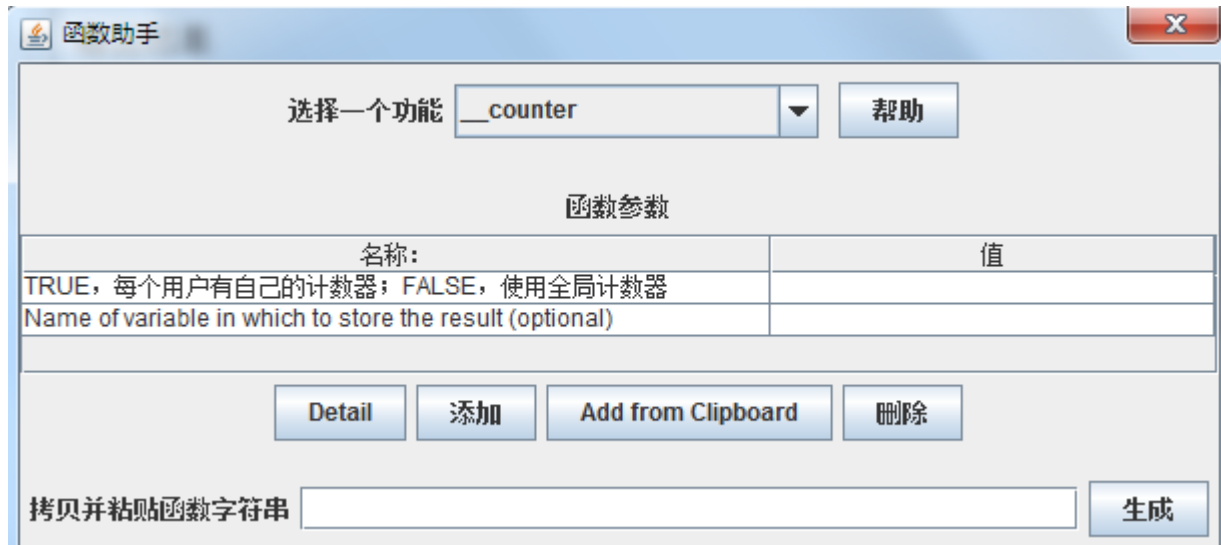
**Name of variable in which to store the result：**存储结果的变量名称（选填）

**Start file sequence number：**初始序列号（选填）

**Final file sequence number：**结束序列号（选填）

### 3.\_Counter 函数

每次调用计数器函数都会产生一个新值，从 1 开始每次加 1。计数器既可以被配置成针对每个虚拟用户是独立的，也可以被配置成所有虚拟用户公用的。如果每个虚拟用户的计数器是独立增长的，那么通常被用于记录测试计划运行了多少遍。全局计数器通常被用于记录发送了多少次请求。计数器使用一个整数值来记录，允许的最大值为 2,147,483,647。



名称:	值
TRUE, 每个用户有自己的计数器; FALSE, 使用全局计数器	
Name of variable in which to store the result (optional)	

【参数】：

第一个参数：True，如果测试人员希望每个虚拟用户的计数器保持独立，与其他用户的计数器相区别。False，全局计数器

第二个参数：重用计数器函数创建值的引用名。测试人员可以这样引用计数器的值：\${count}。这样一来，测试人员就可以创建一个计数器后，在多个地方引用它的值。（选填）

【格式】：\${\_counter(FALSE,count)}

【使用】：我们将“\_counter”函数生成的参数复制到某个参数下面，如果为 TRUE 格式，则每个线程各自统计，最大数为循环数，如果为 FALSE，则所有线程一起统计，最大数为线程数乘以循环数

## 十六、HTTP 属性管理器

Test Plan 的配置元件中有一些和 HTTP 属性相关的元件：HTTP Cache Manager、HTTP Authorization Manager、HTTP Cookie Manager、HTTP Header Manager、 HTTP Request Defaults 等，这些元件有什么作用呢？原因是 JMeter 不是浏览器，因此其行为并不和浏览器完全一致。这些 JMeter 提供的 HTTP 属性管理器用于尽可能模拟浏览器的行为，在 HTTP 协议层上定制发送给被测应用的 HTTP 请求。

### (1) HTTP Request Defaults (HTTP 请求默认值)

该属性管理器用于设置其作用范围内的所有 HTTP 的默认值，可被设置的内容包括 HTTP 请求的 host、端口、协议等。一个 Test Plan 中可以有多个 HTTP Request Defaults，处于多个 HTTP Request Defaults 作用域内的 Sampler 使用 HTTP Request Defaults 中设置值的叠加值。

### HTTP请求默认值

名称:

注释:

**Web服务器**  
服务器名称或IP:  端口号:

**Timeouts (milliseconds)**  
Connect:  Response:

**HTTP请求**

Implementation:  协议:  Content encoding:

路径:

**Parameters**

同请求一起发送参数:

名称:	值	编码?	包含等于?
<div><input type="button" value="Detail"/> <input type="button" value="添加"/> <input type="button" value="Add from Clipboard"/> <input type="button" value="删除"/> <input type="button" value="Up"/> <input type="button" value="Down"/></div>			

**Proxy Server**

服务器名称或IP:  端口号:  用户名:  密码:

**Embedded Resources from HTML Files**

☐ 从HTML文件获取所有内含的资源 ☐ Use concurrent pool. Size:  URLs must match:

## (2) HTTP Authorization Manager

该属性管理器用于设置自动对一些需要 NTLM 验证（NTLM 是 Windows NT 早期版本的标准安全协议）的页面进行认证和登录。

### HTTP授权管理器

名称:

注释:

**Options**  
☐ Clear auth on each iteration?

**存储在授权管理器中的授权**

基础URL	用户名	密码	域	Realm	Mechanism
	51zxw	1234			BASIC_DIGEST

## (3) HTTP Cache Manager (HTTP 缓存管理)

该属性管理器用于模拟浏览器的 Cache 行为。为 Test Plan 增加该属性管理器后，Test Plan 运行过程中会使用 Last-Modified、ETag 和 Expired 等决定是否从 Cache 中获取相应的元素。

**HTTP Cache Manager**

名称: HTTP Cache Manager

注释:

☐ Clear cache each iteration?

☐ Use Cache-Control/Expires header when processing GET requests

Max Number of elements in cache 5000

注意:如果 Test Plan 中的某个 Sampler 请求的元素是被 Cache 的元素,则 Test Plan 在运行过程中会直接从 Cache 中读取该元素,这样 Sampler 得到的返回值就会是空。在这种情况下,如果为该 Sampler 设置了 Assertion 检查响应体中的制定内容是否存在,该 Assertion 就会失败。

#### (4) HTTP Cookie Manager (HTTP Cookie 管理器 )

该属性管理器用于管理 Test Plan 运行时的所有 Cookie。(储存在用户本地终端上的数据)HTTP Cookie Manager 可以自动储存服务器发送给客户端的所有 Cookie,并在发送请求时附上合适的 Cookie.

同时,用户也可以在 HTTP Cookie Manager 中手工添加一些 Cookie,这些被手工添加的 Cookie 会在发送请求时被自动附加到请求。

**HTTP Cookie 管理器**

名称: HTTP Cookie Manager

注释:

**Options**

☐ 每次反复清除Cookies ?

Cookie Policy: rfc2109 Implementation: HC3CookieHandler

**存储在Cookie管理器中的Cookie**

名称:	值	域	路径:	安全
-----	---	---	-----	----

注意: JMeter 的 HTTP Cookie Manager 会为 Thread Group 中的每个线程设置一个单独的会话区域来管理该线程的所有

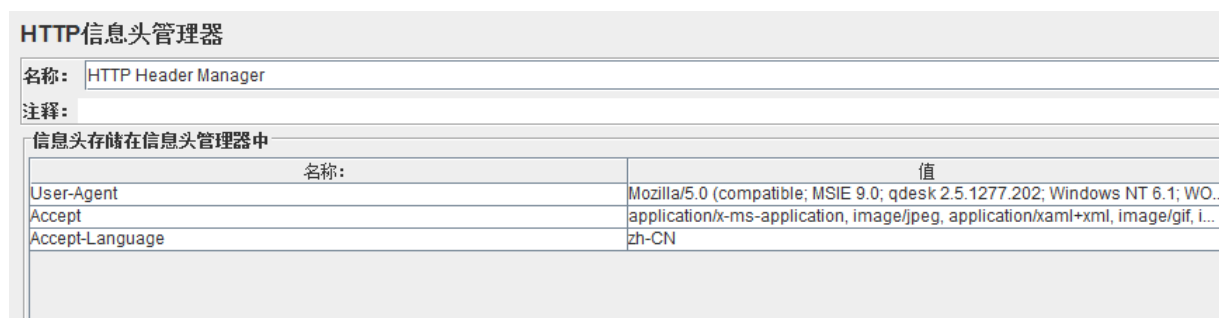


Cookie，也就是说，每个线程的会话 Cookie 是完全独立的（与浏览器行为一致），但用户在 HTTP Cookie Manager 中手工添加的 Cookie 则被所有线程共享。

如果选中"Clear cookies each iteration?"此项，意味着线程在每次迭代时清除自己会话中的所有 Cookie。

## (5) HTTP Header Manager (HTTP 头文件管理器)

该属性管理器用于定制 Sampler 发出的 HTTP 请求的请求头文件的内容。不同的浏览器发出的 HTTP 请求具有不同的 Agent，访问某些有防盗链的页面时需要正确的 Refer...这些情况下都需要通过 HTTP Header Manager 来保证发送的 HTTP 请求是正确的。



名称:	值
User-Agent	Mozilla/5.0 (compatible; MSIE 9.0; qdesk 2.5.1277.202; Windows NT 6.1; WO...
Accept	application/x-ms-application, image/jpeg, application/xaml+xml, image/gif, i...
Accept-Language	zh-CN

# 十七、逻辑控制器

JMeter 中的 Logic Controller 用于为 Test Plan 中的节点添加逻辑控制器。

JMeter 中的 Logic Controller 分为两类：一类用来控制 Test Plan 执行过程中节点的逻辑执行顺序，如：Loop Controller、If Controller 等；另一类则与节点逻辑执行顺序无关，用于对 Test Plan 中的脚本进行分组、方便 JMeter 统计执行结果以及进行脚本的运行控制等，如：Throughput Controller、Transaction Controller。

## 1.控制 Test Plan 中的节点执行顺序的 Logic Controller

### 1.1 ForEach Controller

该逻辑控制器只要用于多个读取自定义变量值。

添加-配置元件-用户自定义变量，定义变量注意命名格式：**变量名+下划线+数字**（从 1 开始 计数）之所以这样命名是为了满足以后 ForEach 控制器提取数据。

## 用户定义的变量

名称: 用户定义的变量

注释:

## 用户定义的变量

名称:	值	Description
user_1	51zxw1	
user_2	51zxw2	
user_3	51zxw3	
user_4	51zxw4	
user_5	51zxw5	

## ForEach控制器

名称: ForEach控制器

注释:

输入变量前缀 user

Start index for loop (exclusive) 1

End index for loop (inclusive) 4

输出变量名称 foreach\_user

☐ Add "\_" before number ?

添加-逻辑控制器-ForEach 控制器

编辑 foreach 控制器

- **变量前缀**: 就是刚才我们定义用户自定义变量下划线前面的字符串
- **取值范围**: 上图表示  $1 \leq \text{user} < 4$ , 表示从数组 user 游标位置为 1 开始取值 (而数据从 0 开始计数), 到游标位置为 4 结束 (但是不包括 5), 所以会从数组变量的第二位开始, 遍历 3 次, 即: 51zxw2, 51zxw3, 51zxw4,
- **输出变量名称**: 我们将每一次得到的用户自定义变量 user 存储到参数 foreach\_user 中

## 1.2 If Controller

类似于编程语言中的 if 语句，根据给定表达式的值决定是否执行该节点下的子节点。

**If Controller**

**Name:** If Controller

**Comments:**

**Condition (default Javascript):** `${_threadNum}>2`

☐ Interpret Condition as Variable Expression? ☐ Evaluate for all children?

注意：

条件判断语句如果是**字符串**一点要在参数前加引号，如： `"${foreach_user}" == "51zxw1"`

如果选中 “Evaluate for all Children” 选项，则该 Controller 在每个子节点执行时执行一次，否则，该 Controller 仅在入口执行一次。 (1.1-1.2 详见脚本 `ForEach.jmx`)

## 1.3 Interleave Controller (交替控制器)

在每次迭代时，顺序选取该节点下的一个子节点执行。

**Interleave Controller**

**Name:** Interleave Controller

**Comments:**

☐ Ignore sub-controller blocks

如果选中 “Ignore sub-controller blocks” 选项，则该 Controller 将其下的子 Controller 当成单一元素处理，并仅允许每个子 Controller 一次发出一个请求。

## 1.4 Loop Controller：(循环控制器)

简单地为其下的子节点运行指定次数。

## 循环控制器

名称: 循环控制器

注释:

循环次数 ☐ 永远 1

### 1.5 Once Only Controller (仅一次控制器)

是为了让 Test Plan 中的某些内容在整个 Test Plan 的执行期间对每个线程仅执行一次（例如，每个线程仅需要执行一次“登录”操作）。如：将 Once Only Controller 作为 Loop Controller 的子节点，Once Only Controller 在每次循环的第一次迭代时均会被执行。（1.3~1.5 参考脚本 Interleave.jmx）

## 仅一次控制器

名称: 仅一次控制器

注释:

### 1.6 Random Controller

## 随机控制器

名称: 随机控制器

注释:

☐ 忽略该控制器块

每次执行时，从其子节点中随机选取一个来执行

### 1.7 Random Order Controller: (随机顺序控制器)

每次执行时，按照随机产生的顺序执行其下的所有子节点。注意，该 Controller 与 Random Controller 的不同之处在于，Random Controller 只选择执行其所有子节点中的一个，而 Random Order Controller 则按照随机顺序全部执行该 Controller 下的所有子节点。

## 1.8 Switch Controller

Switch Controller	
名称:	Switch Controller
注释:	
Switch Value	

类似程序语言中的 switch 函数, 该 Controller 根据给定的值 n(可以使用参数)选择执行其下的第 n+1 个子节点。需要注意的是, Switch Controller 只接受整数值, 并且给定的值 n 是以 0 为基础的, 因此当给定的值 "2" 时, 意味着该 Controller 下的第 3 个子节点会被执行。同时, Switch Controller 接受参数作为其值。

## 1.9 While Controller

该控制器是另一个用于控制循环的 Controller。可以为该 Controller 设置一个 Condition(条件)。其中 Condition 的取值可以为以下三者之一:

While Controller	
名称:	While Controller
注释:	
Condition (function or variable)	LAST

<1>空: 如果 Condition 取值为空, 则该 Condition 迭代执行 Controller 下的所有子节点, 直到最后一个子节点返回失败为止。需要注意的是, 如果不是最后一个子节点失败, 而是该 Controller 中的其他子节点失败, 该 Controller 不会停止, 而会继续循环执行过程。

<2>LAST: 如果 Condition 取值为字符串 LAST, 则该 Controller 迭代执行 Controller 下的所有子节点, 直到最后一个子节点返回失败为止 (与 Controller 为空时的行为相同)。同时, 如果该 Controller 的上一个节点失败, 则 Test Plan 在执行时不会进入该 Controller。

<3>表达式: 除了为空和 LAST 外, Controller 还可以取一个表达式。当 Controller 取值为表达式时, 如果表

达式的取值是字符串 false，While Controller 就退出循环。以下列出了几个可用的表达式：

- ◆ `${VAR}`：当参数 VAR 的值被设置成 false 时退出循环。
- ◆ `${__javascript( "${VAR}" == "51zxw" )}`：当参数 VAR 的值不为 51zxw 时退出循环。

## 2.非控制 Test Plan 中的节点执行顺序的 Logic Controller

### 2.1 事务控制器

**事务控制器**

名称：

事务控制器

注释：

☐ Generate parent sample

☐ Include duration of timer and pre-post processors in generated sample

事务控制器会生成一个额外的采样器来测量其下测试元素的总体时间。值得注意的是，这个时间包含该控制器范围内的所有处理时间，而不仅仅是采样器的。

Attribute	Description	Required
Name	事务控制器的名称	Yes
Sample	如果选中，事务采样器作为其下采样器的父采样器 否则，作为额外采样器添加在子采样器的后面	Yes

【补充说明】对于 Jmeter2.3 以上的版本，有两种模式的操作

- 事务采样器是添加到其下采样器后面的
- 事务采样器是作为其下采样器的父采样器。

生成的事务采样器的测量的时间包括其下采样器以及其他的一切时间。由于时钟频率的问题，这个时间可能略大于单个采样器的时间之和。时钟开始时间介于控制器记录开始时间与第一个采样器开始之间，时钟结束时间亦然。

事务采样器只有在在其子采样器都成功的情况下才显示成功。

在父模式下，事务控制器下的各个采样器只有在 the Tree View Listener 里才能看到。同时，子采样器的数据也不会显示在 CSV 文件中，但是在 XML 文件中可以看到。 1.9~2.1 脚本见 [While\\_Controller.jmx](#)

## 十八 Jenkins+Ant+Jmeter 自动化性能测试平台

### 持续集成

持续集成是一种软件开发实践，即团队开发成员经常集成他们的工作，通常每个成员每天至少集成一次，也就意味着每天可能会发生多次集成。每次集成都通过自动化的构建（包括编译，发布，自动化测试）来验证，从而尽快地发现集成错误。许多团队发现这个过程可以大大减少集成的问题，让团队能够更快的开发内聚的软件。

Jenkins 是基于 Java 开发的一种持续集成工具，用于监控持续重复的工作，功能包括：持续的软件版本发布/测试项目，监控外部调用执行的工作。

### 项目构建

通过构建工具对多个项目文件进行统一批量的编译和运行。比如，对多个 Jmeter 脚本批量运行。

Apache Ant，是一个将软件编译、测试、部署等步骤联系在一起加以自动化的一个工具，大多用于 Java 环境中的软件开发。

### Ant 安装配置

#### 1. 下载安装

下载地址：<http://ant.apache.org/bindownload.cgi>

**Mirror**

You are currently using <http://apache.fayea.com/>. If you encounter a problem with this mirror, please see *backup mirrors* (at the end of the mirrors list) that should be available.

Other mirrors:

**Current Release of Ant**

Currently, Apache Ant 1.9.4 is the best available version, see the [release notes](#).

**Note**

Ant 1.9.4 was released on 05-May-2014 and may not be available on all mirrors for a few days.

**Tar files may require gnu tar to extract**

Tar files in the distribution contain long file names, and may require gnu tar to do the extraction.

- zip archive: [apache-ant-1.9.4-bin.zip](#) [PGP] [SHA1] [SHA512] [MD5]
- tar.gz archive: [apache-ant-1.9.4-bin.tar.gz](#) [PGP] [SHA1] [SHA512] [MD5]
- tar.bz2 archive: [apache-ant-1.9.4-bin.tar.bz2](#) [PGP] [SHA1] [SHA512] [MD5]

**Old Ant Releases**

下载解压放在任意盘符，随后进行环境变量配置（windows 为例）

ANT\_HOME C:/ apache-ant-1.9.0  
path C:/ apache-ant-1.9.0/bin  
classpath C:/apache-ant-1.9.0/lib

在 DOS 界面输入如下命令： {Ant 安装位置} \bin ant -version

如果出现如下内容，说明安装成功：

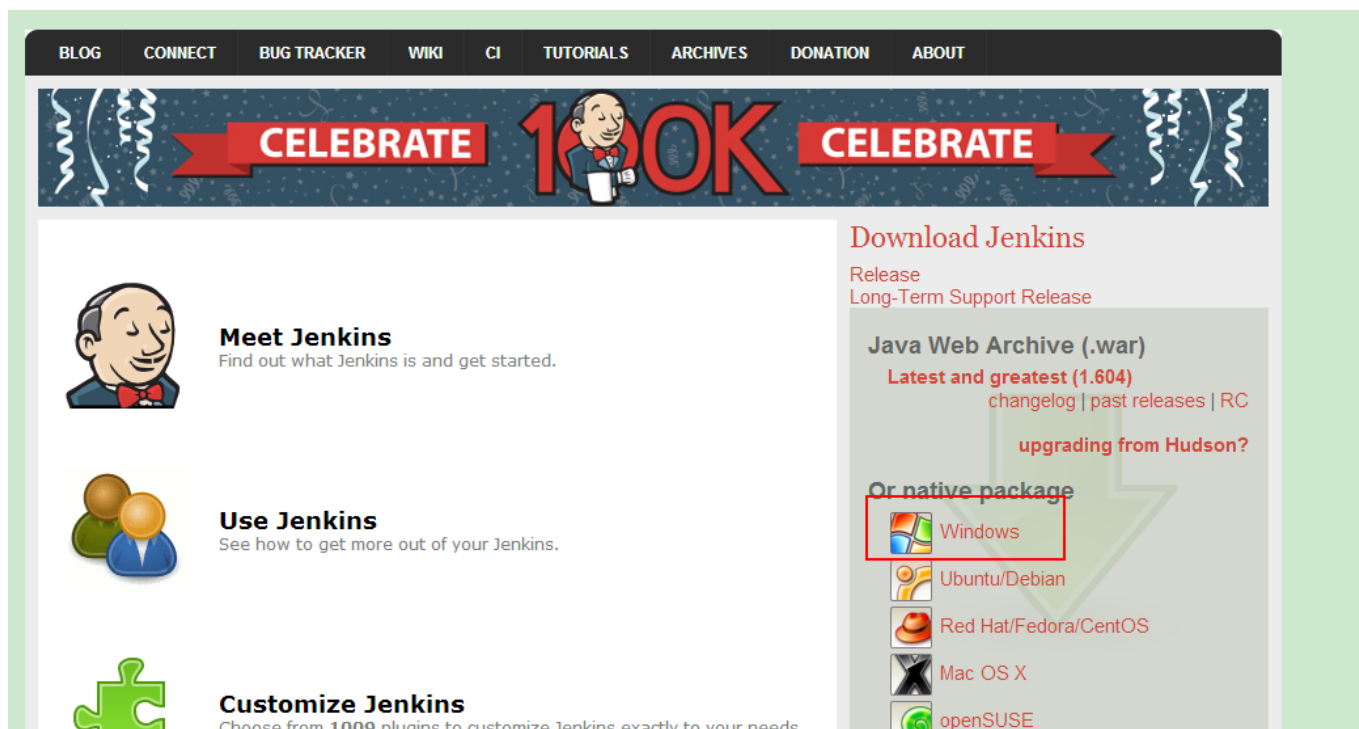
```
c:\ant\bin>ant -version
Apache Ant(TM) version 1.9.4 compiled on April 29 2014
```

说明 ant 安装成功！但如果出现 'ant' 不是内部或外部命令，也不是可运行的程序或批处理文件。说明安装失败

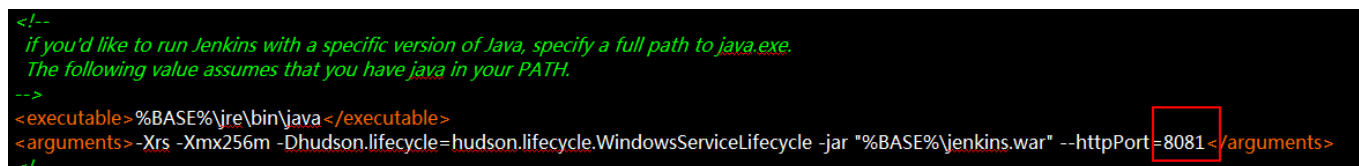
## Jenkins 安装配置

Jenkins 下载：<http://jenkins-ci.org>

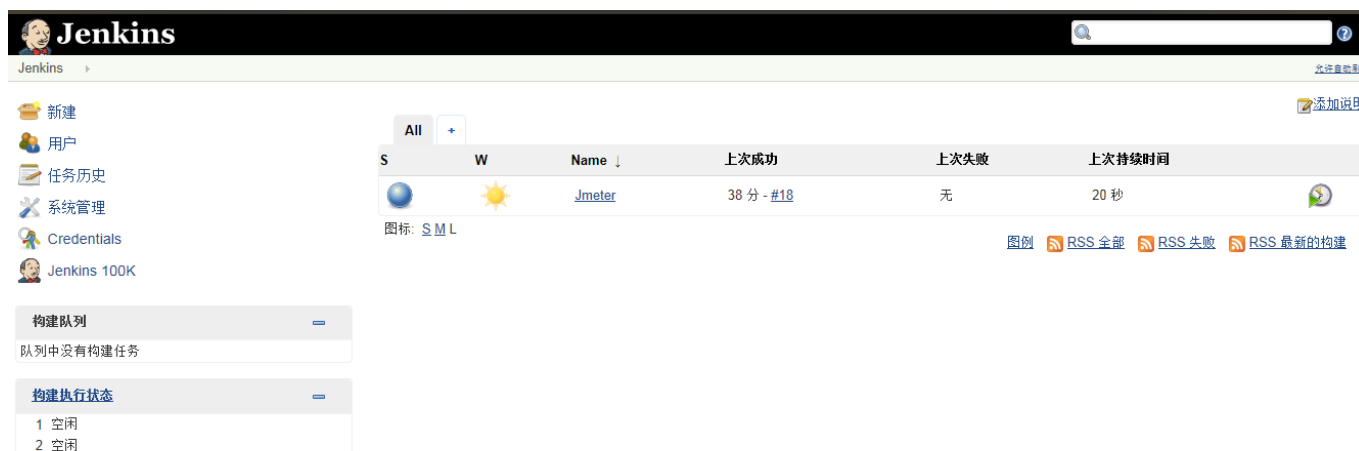




解压后放在一个磁盘空间大的盘符, 然后点击 setup.exe, 安装完成后在浏览器输入: `http://localhost:8080` (默认 8080 端口) 如果 8080 端口被占, 修改 jenkins 安装目录下的配置文件: `jenkins.xml` 中的 `httpPort=8080`, 如下图, 我已经改成了 8081 端口号。



浏览器输入制定地址后显示如下界面则为 jenkins 安装成功。



## Jenkins+Ant+Jmeter 自动化测试平台搭建

操作步骤:

**Step1:**录制 jmeter 脚本 (此处省略 500 字。。。不会的参考本套视频 2-6)

**Step2:**将 Jmeter 的安装目录下的 extras 目录中 “ant-jmeter-1.1.1.jar” 这个 jar 包放到 ant 的 {安装目录} \lib 目录下 (此处很关键!) 还有 jmeter-results-detail-report\_21 复制到 ant 下面 bin 目录

**Step3:**配置 Jemter 的 build.xml 配置文件。然后放到 ant 下面 bin 目录

### Jmeter Ant xml 配置文件

#### 1.Ant 关键元素

Ant 的构件文件是基于 XML 编写的, 默认名称为 build.xml

**project** 元素是 Ant 构件文件的根元素, Ant 构件文件至少应该包含一个 project 元素, 否则会发生错误。在每个 project 元素下, 可包含多个 target 元素。接下来熟悉 project 元素的各属性。

##### 1) name 属性

用于指定 project 元素的名称。

##### 2) default 属性

用于指定 project 默认执行时所执行的 target 的名称。

##### 3) basedir 属性

用于指定 jmeter 基路径的位置。该属性没有指定时, 使用 Ant 的构件文件的附目录作为基准目录。

#### 2.target 元素

它为 Ant 的基本执行单元, 它可以包含一个或多个具体的任务。多个 target 可以存在相互依赖关系。它有如下属性:

下属性:

##### 1) name 属性

指定 target 元素的名称，这个属性在一个 project 元素中是唯一的。我们可以通过指定 target 元素的名称来指定某个 target。

## 2) depends 属性

用于描述 target 之间的依赖关系，若与多个 target 存在依赖关系时，需要以 “,” 间隔。Ant 会依照 depends 属性中 target 出现的顺序依次执行每个 target。被依赖的 target 会先执行。

## 3) if 属性

用于验证指定的属性是否存在，若不存在，所在 target 将不会被执行。

## 4) unless 属性

该属性的功能与 if 属性的功能正好相反，它也用于验证指定的属性是否存在，若不存在，所在 target 将会被执行。

## 5) description 属性

该属性是关于 target 功能的简短描述和说明。

## 3.property 元素

该元素可看作参量或者参数的定义，project 的属性可以通过 property 元素来设定，也可在 Ant 之外设定。若要在外部引入某文件，例如 build.properties 文件，可以通过如下内容将其引入：

```
<property file= "build.properties" />
```

**Step4.**使用 Ant 编译验证 Jmeter 的 build 文件。(注意将配置好的 build.xml 文件放在 ant 目录的 bin 目录下)

**Step5:**部署到持续集成平台 jenkins.(Game over,此处应该有掌声! )



我要自学网性能测试进阶视频教程

文档参考资料（感谢这些默默无私奉献的小伙伴们~）

1. <http://www.51testing.com/>
2. <http://www.cnblogs.com/yuyii/>
3. <http://www.cnblogs.com/fnng/>