

目錄

简介	1.1
认识JMeter	1.2
了解并启动JMeter	1.2.1
认识JMeter	1.2.2
第一个测试	1.2.3
提高JMeter	1.3
线程高并发	1.3.1
逻辑控件器	1.3.2
断言测试	1.3.3
结果分析	1.3.4
完善JMeter	1.4
HTTP信息头管理	1.4.1
数据获取	1.4.2
实战性能测试	1.4.3
分析测试	1.5
断言结果详细分析	1.5.1
结果详细分析	1.5.2
服务器分析	1.5.3
扩展插件	1.5.4
进阶使用	1.6
命令行执行JMeter	1.6.1
性能测试常用专业术语	1.6.2
JMeter最佳实践	1.6.3
性能测试最佳实践	1.6.4
测试报告	1.6.5
结束语	1.7

本文目的

- 独立完成项目的性能测试
- 快速分析并使用**JMeter**定位出项目性能测试结果
- 产出有价值的性能测试报告

目录

- 认识**JMeter**: 了解并认识**JMeter**功能
- 提高**JMeter**: 项目中常用的**JMeter**功能
- 完善**JMeter**: 丰富**JMeter**在项目中的具体使用
- 分析测试: 针对已有的测试结果进行服务器/客户端分析，并引入扩展插件进行完善的结果展示
- 进阶使用: 项目中如何高效识别出哪些业务需要测试，如何让**JMeter**的性能发挥的最好，**JMeter**与持续集成的结合使用，性能测试报告的注意事项
- 结束语: 建议&End

说明

- 文中使用的**JMeter**版本：**JMeter 3.0**
- 文中关于**JMeter**的功能，不会很详尽，够用足以

认识JMeter

- 了解并启动JMeter
 - JMeter是什么，用JMeter可以做什么
 - 安装JMeter，启动JMeter的几种方式
- 认识JMeter
 - 全面了解JMeter-UI的功能
- 第一个测试
 - 中文版本的使用
 - 使用JMeter完成第一个测试

了解**JMeter**

JMeter是什么

- 开源软件，纯免费
- 100%基于Java开发的
- 完美的UI界面
- 支持多线程并发测试
- 完善的可视化测试结果
- 各种插件及自定义插件

JMeter可以用来测试哪些

支持各种服务及协议类型

- Web - HTTP, HTTPS
- SOAP / REST
- FTP
- Database
- LDAP
- Message-oriented middleware (MOM)
- Mail
- TCP
- Native commands / shell scripts

启动**JMeter**

前提准备

- 安装 Java7 或更新的版本
- 配制完成Java的环境变量

安装**JMeter**

JMeter官方版本是免安装的，只需要从官网获取压缩包即可。

- 打开http://jmeter.apache.org/download_jmeter.cgi页面
- 下载 Binaries 中的**apache-jmeter-3.0**，tgz或zip格式均可
- 解压下载的压缩包，安装JMeter完成

启动JMeter

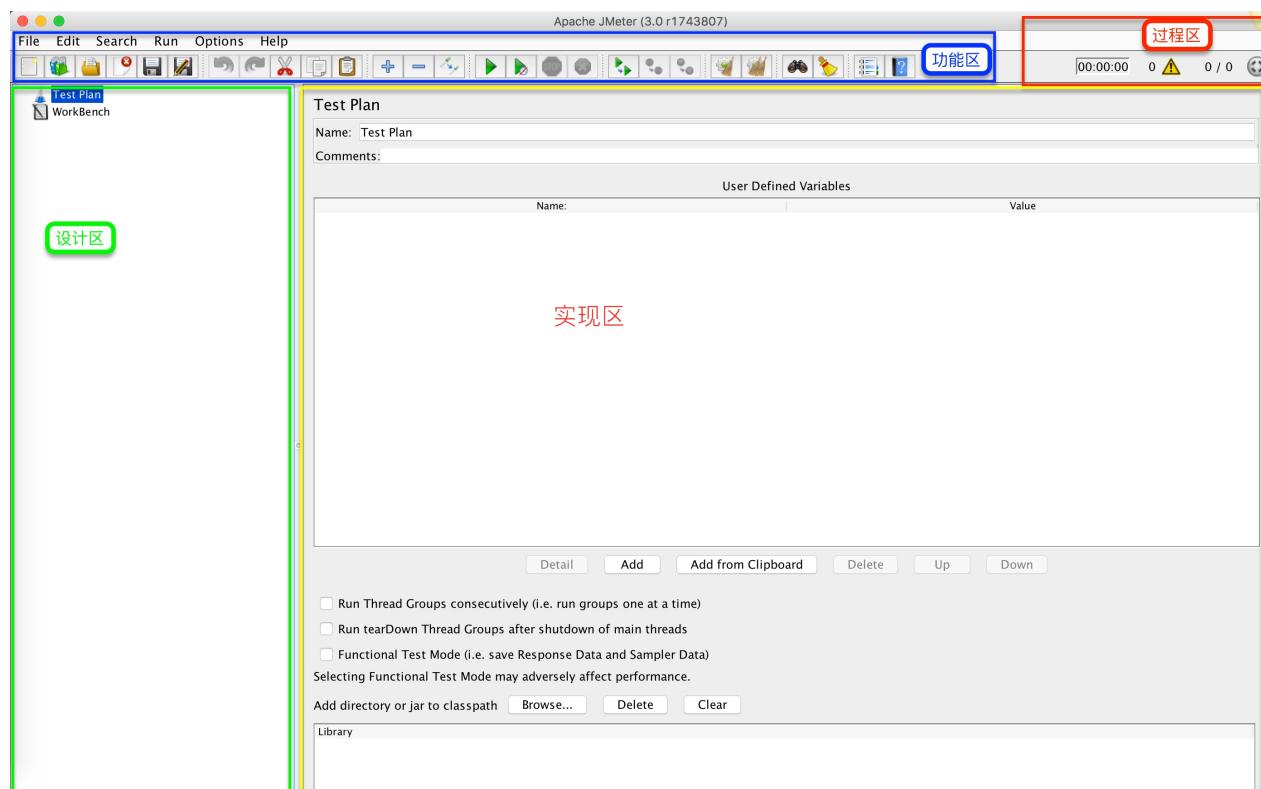
- 进入解压文件夹的**apache-jmeter-3.0/bin** 目录
- 启动方式
 - MAC 电脑启动
 - 运行 jmeter
 - Windows 电脑启动
 - 双击 jmeter.bat
 - 命令行 启动: java -jar ApacheJMeter.jar
- 启动成功后，便可看到**JMeter**的UI界面

JMeter 目录分析



目录及对应的文件的使用，在后续内容中会详细讲解。

界面

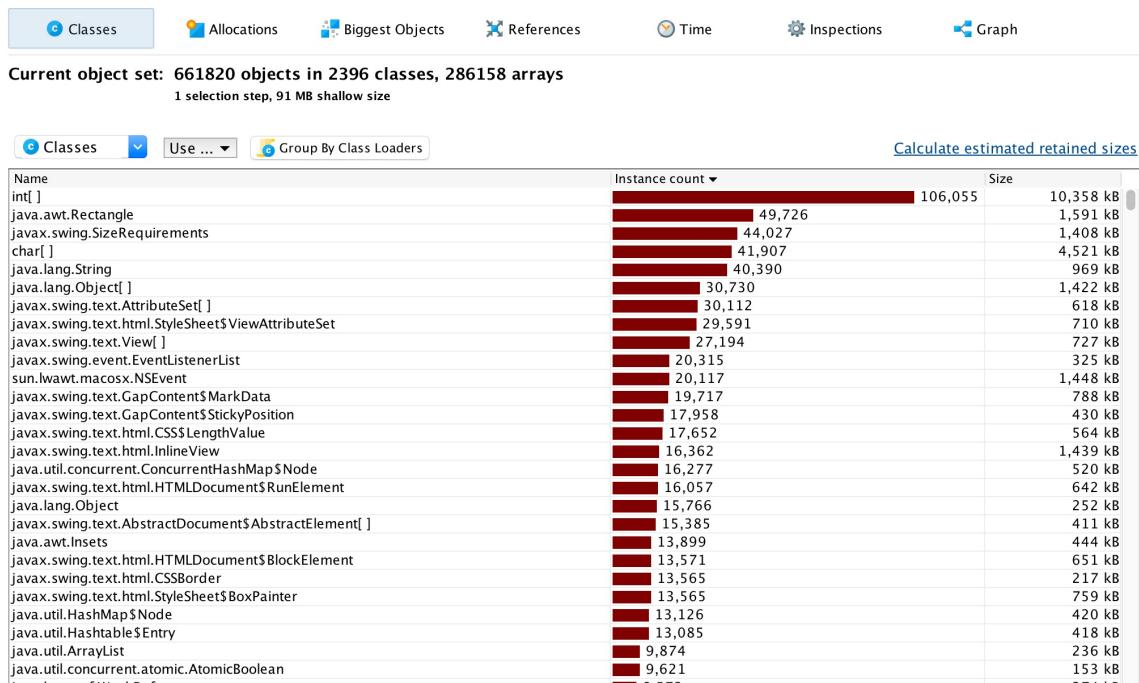


功能区

主要用于界面样式/启动/运行配制相关。

菜单区

- File: 文件操作/保存等，如：新建测试计划/保存测试计划
 - Open: 打开一个已有的 测试计划
 - Templates: 使用已有的 模板 ，快速创建 测试计划
 - Merge: 将当前打开的测试计划与目标测试计划合并
- Edit: 测试计划的编辑，如：新增请求/添加监听器。内容依据选择的 节点 不同而变化
 - Add: 依据当前选中的节点，进行添加新节点操作
 - Reset GUI: 重置当前 实现区 的配制内容
 - Enable/Disable: 启用/禁用选中的节点
- Search: 计划中内容查找
 - Search: 搜索节点名
 - Reset Search: 清空Search结果
- Run: 执行测试，如：本地执行/远程执行
 - Start*: 执行本地的测试计划
 - Remote*: 远程操作
 - Clear: 清空当前选中的执行结果
 - Clear All: 清空所有的执行结果
- Options: 样式/展示效果设置
 - Function Helper Dialog: 弹出方法的帮助说明，可查看具体的使用
 - Look and Feel: 样式选择
 - Log Viewer: 执行测试计划过程中的日志显示选项，勾选时，显示执行过程中的日志内容
 - SSL: 选择执行过程中所需要的SSL加密文件
 - Choose Language: 评论选择
 - Collapse/Expand all: 收起/展开所有的节点
- Help: 帮助相关
 - Help: 直接打开当前 实现区 的帮助内容，并定位到对应的位置
 - What's this node?: 定位到当前节点在 设计区 的位置
 - Enable/Disable debug: 开启/关闭Debug模式
 - Create a heap dump: 创建当前程序的Dump文件，用于查看资源的使用情况



快速操作区

将 菜单区 中常用的功能拉到下面，看提示信息就能看懂啦

设计区

- 测试计划如何配制
- 测试用例如何设计
 - 测试前提准备
 - 测试步骤
 - 测试结果验证
- 测试场景如何组装
- 测试结果如何查看

实现区

针对 设计区 的内容进行细节的配制及实现。如：测试结果的验证/测试步骤的执行

过程区

测试过程中的数据展示，执行时间/错误数

JMeter语言支持

JMeter 支持 英语/法语/德语/挪威语/波兰语/葡萄牙语/西班牙语/土耳其语/日语/中文简体/中文繁体

UI语言切换- 简体中文

- Options -> Choose Language -> Chinese (Simplified)

说明

- JMeter的语言包翻译的不够完善，有些功能甚至还是未翻译的。在使用过程中，最好使用默认 英文版本

第一个测试

功能说明

- 访问请求<http://www.jianshu.com/>
- 查看返回状态码为200

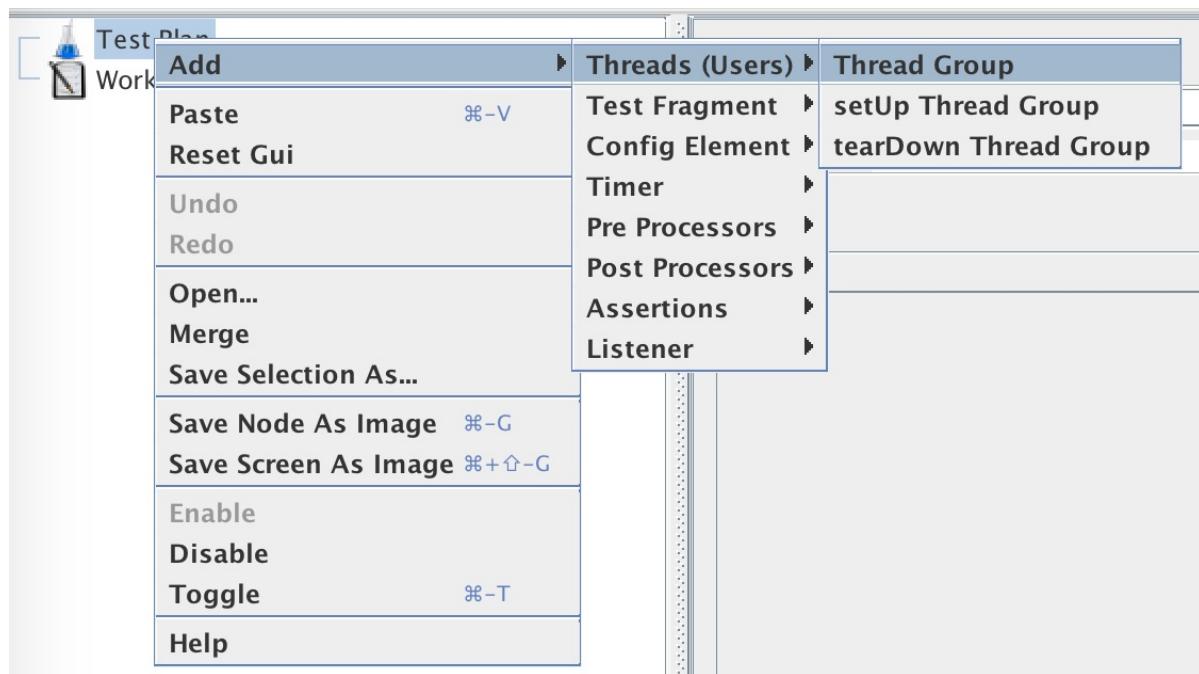
设计测试

添加测试计划--Test Plan

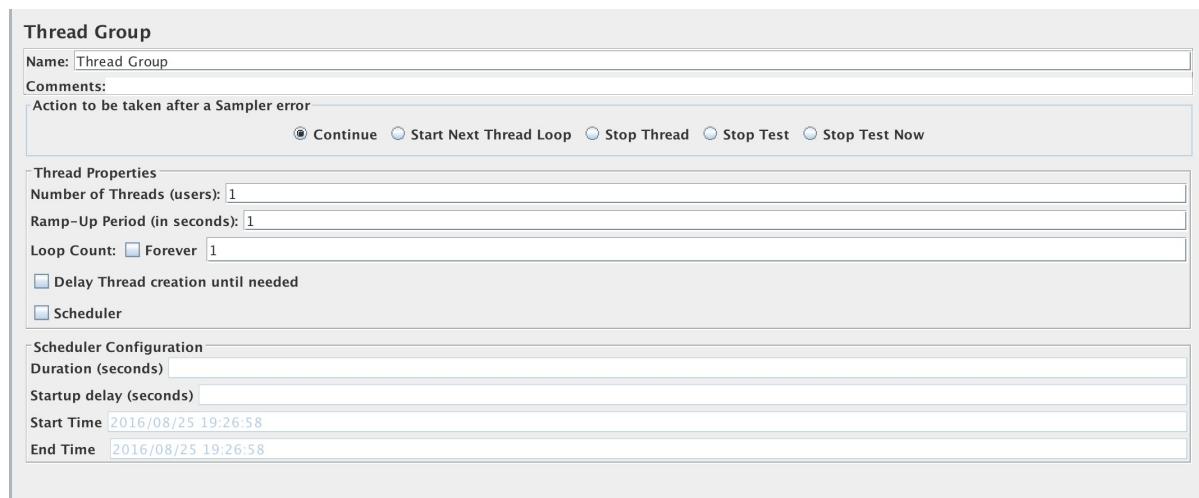
在打开JMeter后，会默认有一个空白内容的测试计划。如果没有，点击 功能区 -- New 按钮

添加线程组--Thread Group

- 右键点击 Test Plan ，选择 Add - Threads(Users) - Thread Group

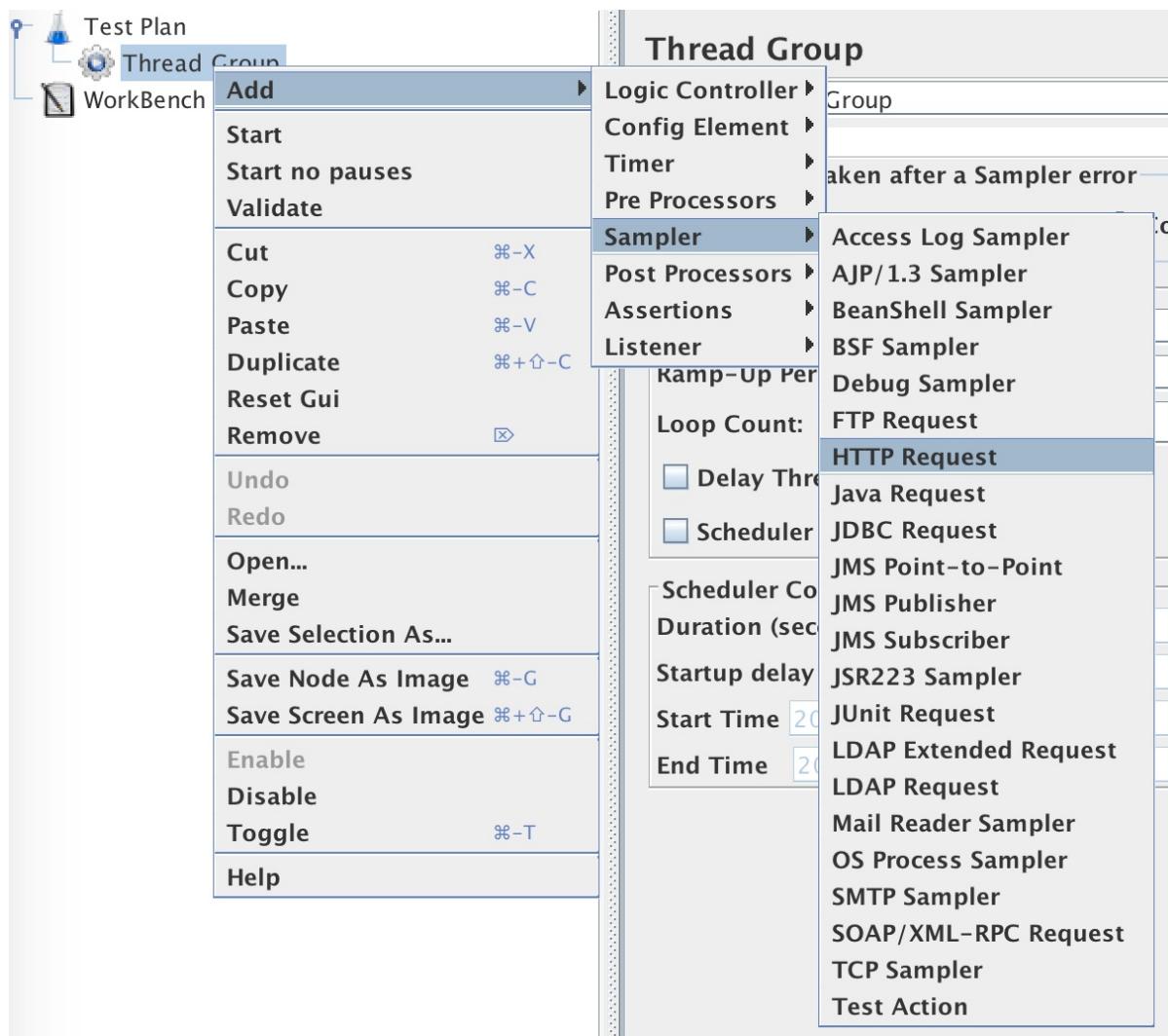


- 在 实现区 会出现线程组的实现配制

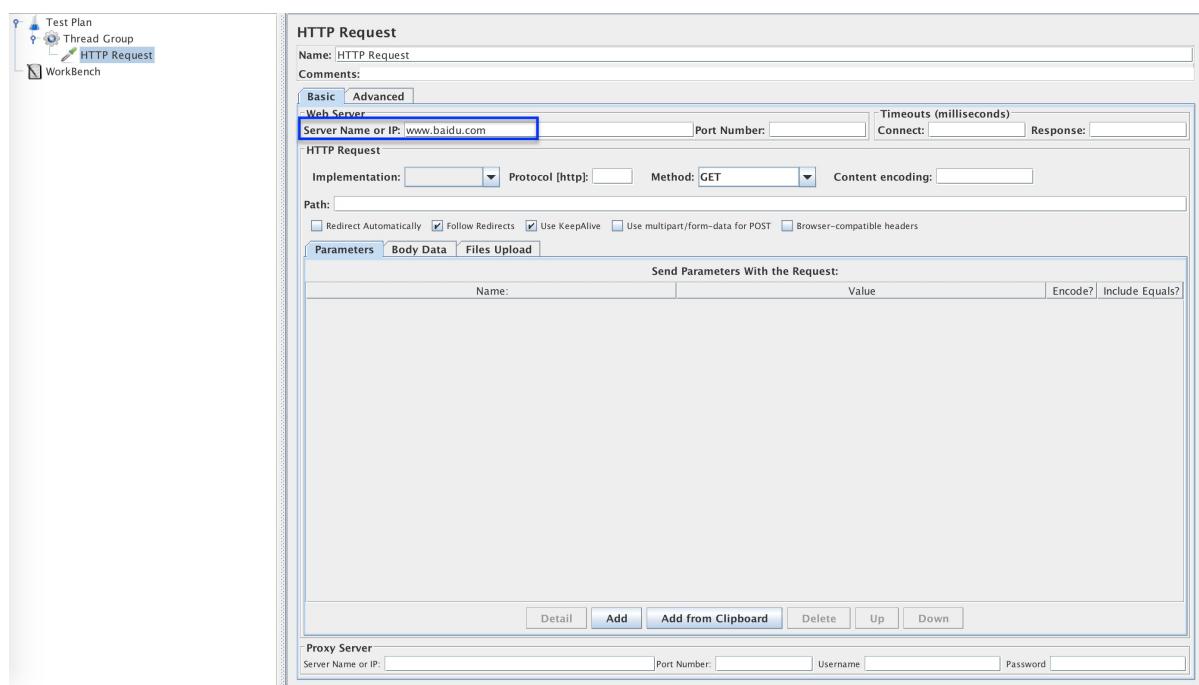


添加HTTP请求--HTTP Request

- 右键点击 Thread Group , 选择 Add - Sampler - HTTP Request

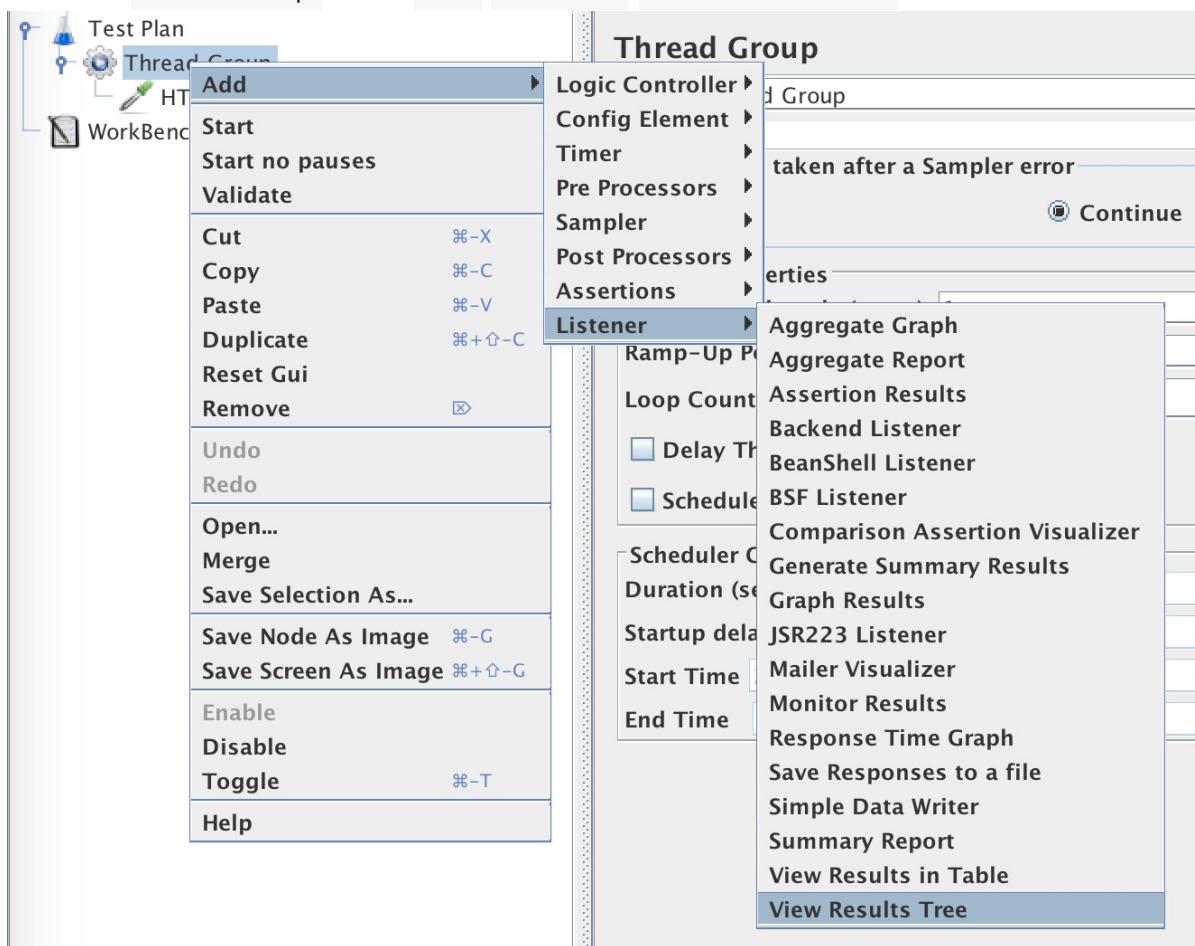


- 在 实现区 会出现HTTP请求的实现配制
- 配制HTTP的 Basic 属性中的 Server Name or IP : www.jianshu.com

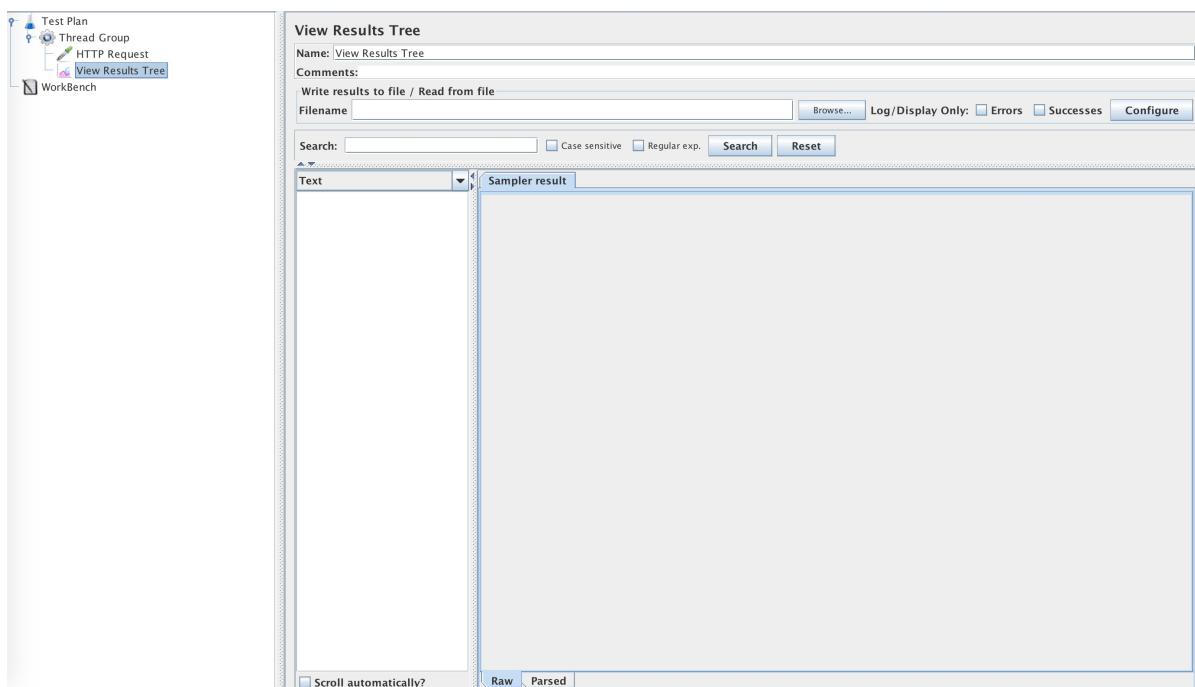


添加结果查看--Result Tree

- 右键点击 Thread Group , 选择 Add - Listener - View Results Tree



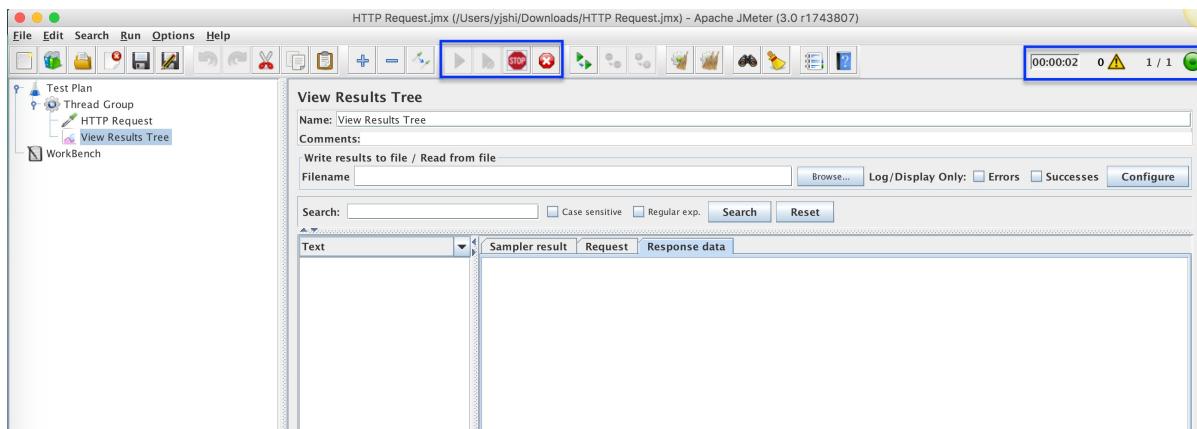
- 在 实现区 会出现Result Tree的实现配制，此处暂时不用做任何配制



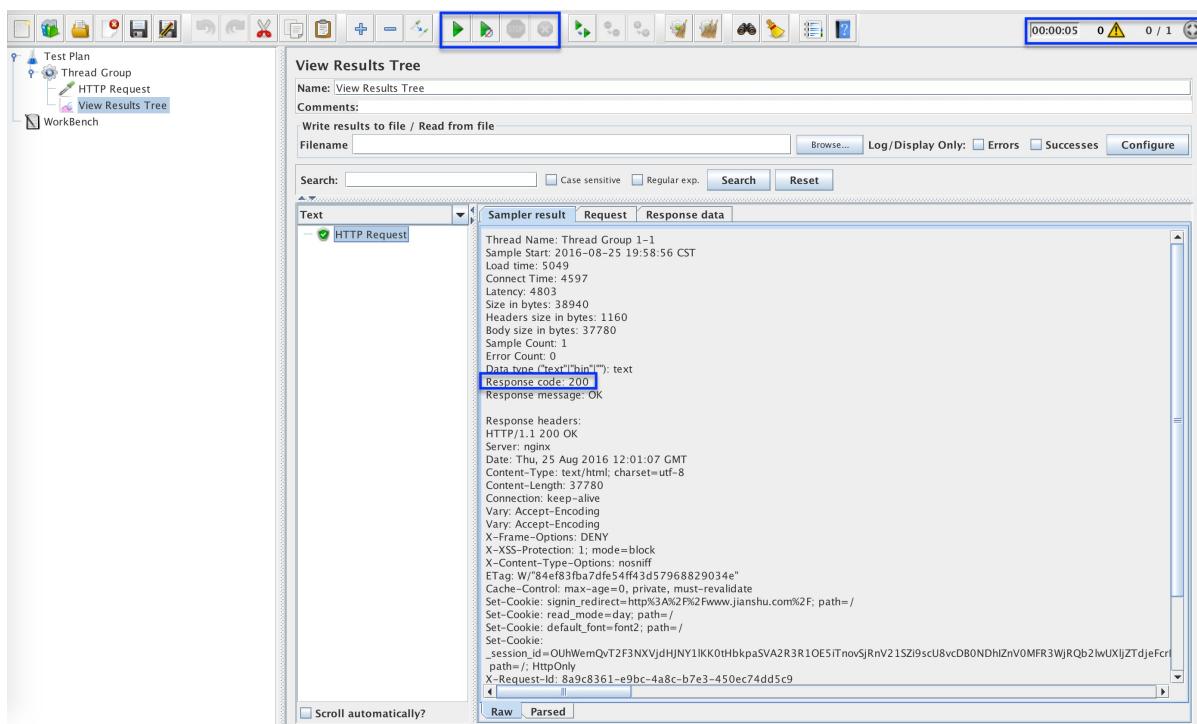
测试设计完成，保存 *Test Plan*

执行测试

- 点击功能区中的 Start 按钮后，可看到 Start 按钮被禁灰 / Stop / Shutdown 按钮进入可点击状态
- 过程区会显示执行测试的过程数据



- 执行结束后，在 Result Tree 会显示本次的测试结果。点击 Result Tree 中的 HTTP Request，此时会显示出本次请求的结果明细



- 返回状态码 Response Code:200

至此，第一个 JMeter 的测试请求完成。

完整的源文件存放于 [HTTPRequest.jmx](#)

提高JMeter

- 线程高并发
 - 测试过程中如何使用JMeter提高并发
- 逻辑控件器
 - 详解JMeter自带 17 种控件器，满足测试场景中的各种需求
- 断言测试
 - 常用的断言方式
 - 断言实例演示
- 结果分析
 - 结果树 展示每次测试过程中的数据，并且支持各种形式查看结果
 - JMeter自带的 Summary Report 测试结果分析

线程高并发

测试过程中，往往需要提高测试的并发数量，使用[JMeter](#)可很快速的完成此功能。

线程组分类

- `setUp Thread Group` 前置线程组，主要用于在执行测试前的一些工作。如清除数据/注册账号。
- `Thread Group` 真实执行的测试线程组。实现性能测试所需的所有内容。
- `tearDown Thread Group` 销毁线程组，主要用于测试结束后的收尾工作。如清除测试脏数据/环境还原。

配制线程组

The screenshot shows the 'Thread Group' configuration dialog in JMeter. It includes fields for Name (Thread Group), Comments, and an 'Action to be taken after a Sampler error' section with radio buttons for Continue (selected), Start Next Thread Loop, Stop Thread, Stop Test, and Stop Test Now. Below these are sections for 'Thread Properties' (Number of Threads: 1, Ramp-Up Period: 1 second, Loop Count: Forever 1, Delay Thread creation until needed, Scheduler), 'Scheduler Configuration' (Duration: 0 seconds, Startup delay: 0 seconds), and 'Timing' (Start Time: 2016/09/01 13:19:12, End Time: 2016/09/01 13:19:12).

- `name` : 用户线程组的命名
- `Comments` : 线程组的功能的备注，可为空
- `Action to be taken after a Sampler error` : 样式错误后采取的方案
 - `Continue` : 继续执行
 - `Start Next Thread Loop` : 结束当前Thread，直接开启下一个新的线程
 - `Stop Thread` : 停止线程
 - `Stop Test` : 在当前样例结束后，停止测试
 - `Stop Test Now` : 马上 强制 停止测试
- `Thread Properties` : 线程属性

- Number of Threads(users) : 需要模拟的用户/线程数
- Ramp-Up Period(in seconds) : JMeter 多久将所有的线程启动。
- Loop Count : 循环数
 - Forever : 选择后，测试只能通过 手工 停止;否则会一直执行
 - 配制具体的循环数
- Delay Thread creation until needed : 此选择仅在 Thread Group 中有。延迟线程仅在需要的时候创建
- Scheduler : 勾选后，才可配制以下内容
 - Duration(seconds) : 测试持续时间
 - Startup delay(seconds) : 开始延迟时间
 - Start Time : 指定测试开始时间
 - End Time : 指定测试结束时间

应用

测试过程中，需要有效的提升 JMeter 并发，主要围绕在 Number of Threads(users) / Ramp-Up Period(in seconds) / Loop Count 的配制中。

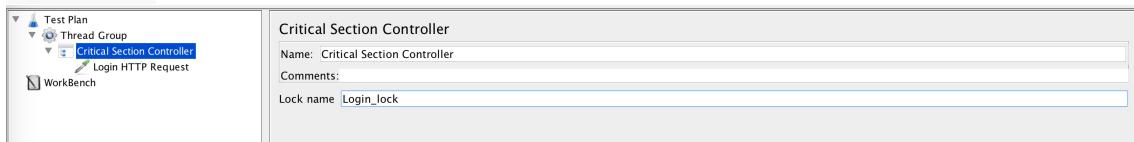
逻辑控件器

逻辑控制器: 请求过程中，需要一些进行逻辑处理的控制器。如: 用户必须是登录成功后，才可访问个人信息页面

Critical Section Controller

- 关键部分控制器，用于核心部分的控制，但此部分的内容在一个线程中仅会执行一次
- 应用场景: 用户登录
- 配制说明

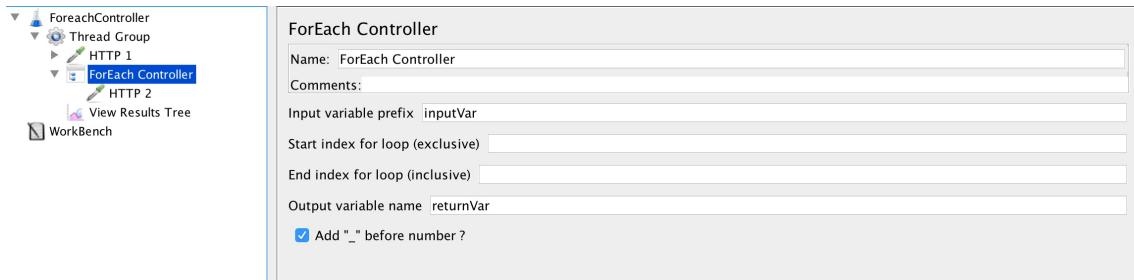
- Lock name : 仅配制名，不启实际作用



- 示例代码: [CriticalSectionController.jmx](#)

Foreach Controller

- 遍历变量中的所有值
- 应用场景: 遍历变量的所有值。如: 获取用户ID列表，再依次查看用户用户信息
- 配制说明
 - Input variable prefix : 需要遍历的变量
 - Start index for loop(exclusive) : 从哪个索引开始，不填表示从第一个元素开始
 - End index for loop(inclusive) : 从哪个索引结束，不填表示遍历至变量内容的最后一个元素
 - Output variable name : 遍历出来结果的变量，在控制器中使用的变量名
 - Add "_" before number ? : 在数字前添加 _，默认勾选。如果是使用正则表达式提取的变量，一定要勾选此项



- 示例参考: [ForEachController.jmx](#)

If Controller

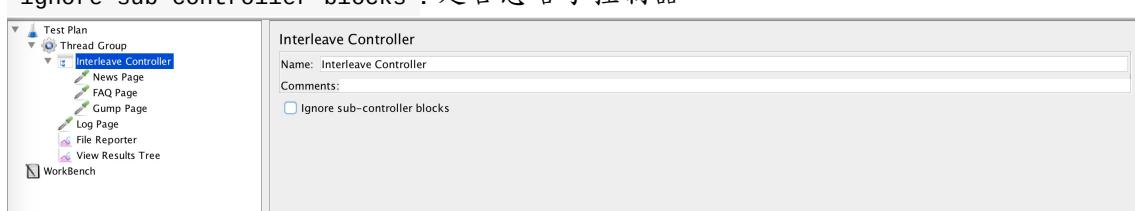
- If 状态判断控制器
- 配制说明
 - Condition(default Javascript) : If 的判断条件，默认支持 Javascript 语法
 - Interpret Condition as Variable Expression? : 勾选后，表达式的结果要为 真 时，才算判断成功。
 - Evaluate for all children? : 是否对所有的子控制器进行校验
- 示例参考: [IfController.jmx](#)

Include Controller

- 引用外部 测试计划 控制器
- 应用场景: 测试过程中，需要引用外部的 测试计划
- 配制说明
 - Include Test Plan-Filename : 选择需要引入的外部测试计划文件
- 示例参考: [IncludeController.jmx](#)

Interleave Controller

- 间隔控制器，每个 线程 用户仅执行一次控制器内的请求， 线程 用户依据循环的次数请求控制器中的请求数
- 配制说明
 - ignore sub-controller blocks : 是否忽略子控制器



- 示例代码: [Interleave.jmx](#)

Loop Controller

- 循环控制器
- 应用场景: 指定循环次数或无限次循环
- 配制说明

- Loop Count : `Forever` 勾选后，则控制器一直循环；不勾选时，可直接设置循环的次数

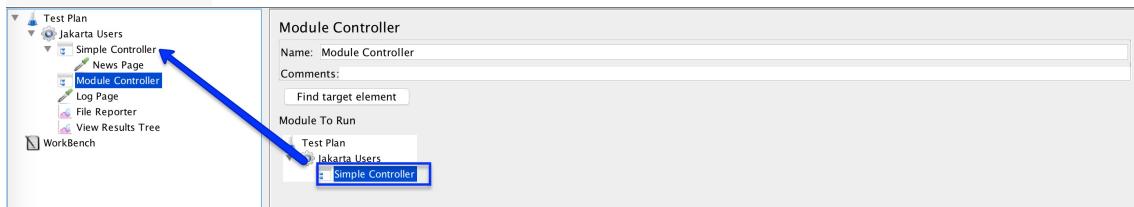


- 示例代码: [LoopController.jmx](#)

Module Controller

- 模块控制器，用于跳转到选定的控制器位置并执行对应的控制器
- 应用场景: 业务逻辑的跳转
- 配制说明

- Module to Run : 选择需要跳转到的目标控制器

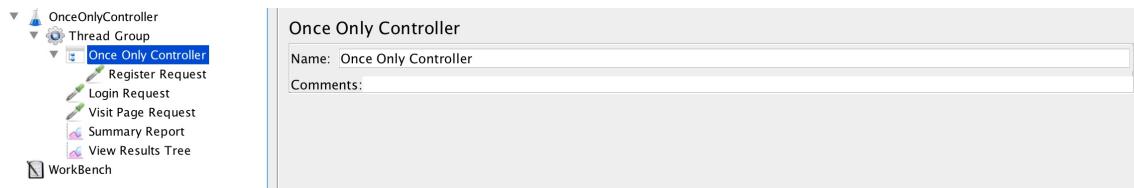


- 示例代码: [ModuleController.jmx](#)

Once Only Controller

- 每个线程用户仅运行一次
- 应用场景: 在线程中仅需要运行一次，后续不需要再次运行。如：创建用户
- 配制说明

- 控制器不额外配制，将需要执行一次的内容放到控制器内，即可

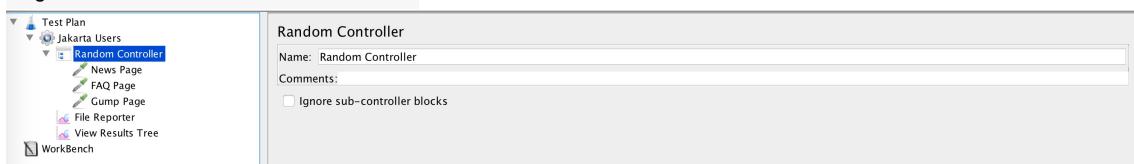


- 示例参考: [OnceOnlyController.jmx](#)

Random Controller

- 随机控制器，随机选择控制器中的请求进行执行
- 应用场景: 页面的随机访问
- 配制说明

- ignore sub-controller blocks : 忽略子控制器

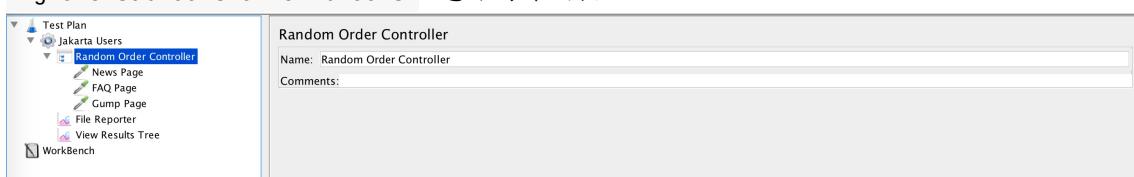


- 示例代码: [RandomController.jmx](#)

Random Order Controller

- 随机顺序执行，与 Random Controller 不同的是，这个控制器会先将需要随机的内容均执行一遍，但次序不定
- 应用场景: 页面的随机访问,但均需要访问，且次序不限
- 配制说明

- ignore sub-controller blocks : 忽略子控制器



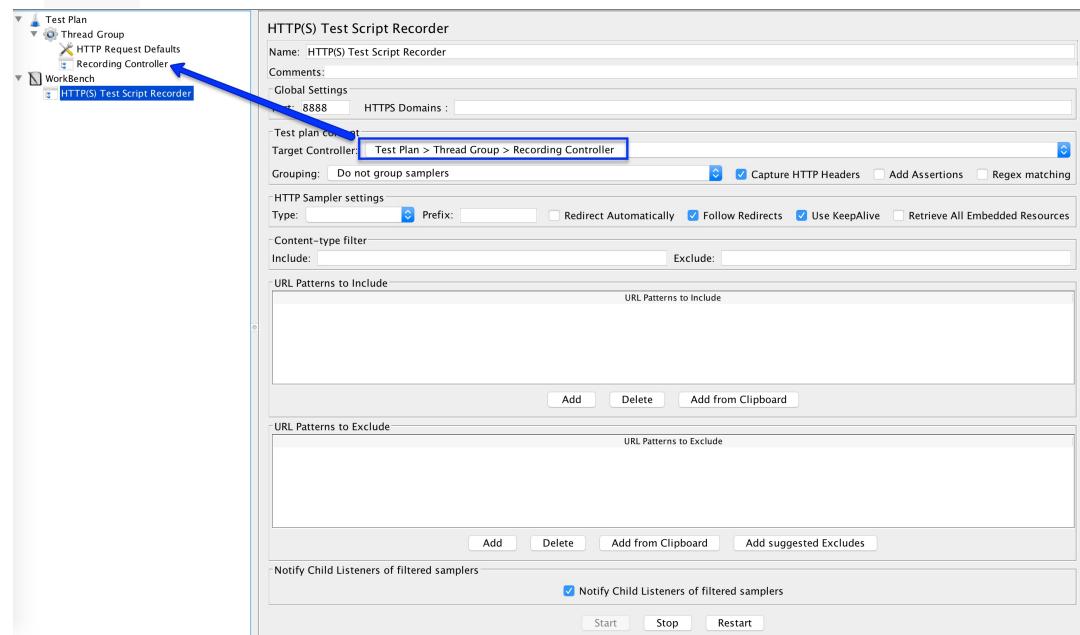
- 示例代码: [RandomOrderController.jmx](#)

Recording Controller

- 录入控制器，直接使用JMeter的代理来录入 浏览器 的请求信息
 - 录入时，需要设置浏览器的 代理地址及端口
- 应用场景: 针对浏览器的请求信息进行 录入

- 配制说明

- Recording Controller 自身不需要配制，但需要 HTTP(S) Test Script Recorder 和 HTTP Request Defaults 的配制
- HTTP Request Defaults：配制 Web Server-Server Name or IP 中配制需要录入的目标地址的根URL
- HTTP(S) Test Script Recorder
 - Port：代理所需要使用的端口号
 - Test plan content--Target Controller：录入内容需要存放的位置
 - Test plan content--Grouping：录入内容的分组策略
 - URL Patterns to Include：过滤需要录入的URL地址，支持正则表达式
 - URL Patterns to Exclude：过滤不需要录入的URL地址，支持正则表达式
 - 设置浏览器的代理地址及端口，点击 HTTP(S) Test Script Recorder 下方的 Start 后，在浏览器中输入地址，即可开始录入



- 示例代码：[RecordingController.jmx](#)

Runtime Controller

- 运行时间控制器，设置控制器中的请求运行时间，单位：秒
- 应用场景：对业务请求时间进行限制
- 配制说明

- Runtime(seconds)：配制需要设置的运行时间，单位 秒

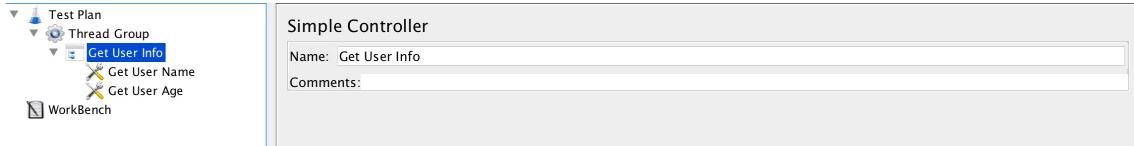


- 示例代码: [RuntimeController.jmx](#)

Simple Controller

- 简单控制器，不提供任何 逻辑功能
- 应用场景: 请求的管理
- 配制说明

- 此控制器不需要配制



- 示例代码: [SimpleController.jmx](#)

Switch Controller

- 跳转控制器，类似编程语言中的 Switch 语法功能
- 应用场景: 满足条件执行的控制器
- 配制说明

- Switch Value : 配制需要跳转第几个的元素，元素值从0开始



- 示例代码: [SwitchController.jmx](#)

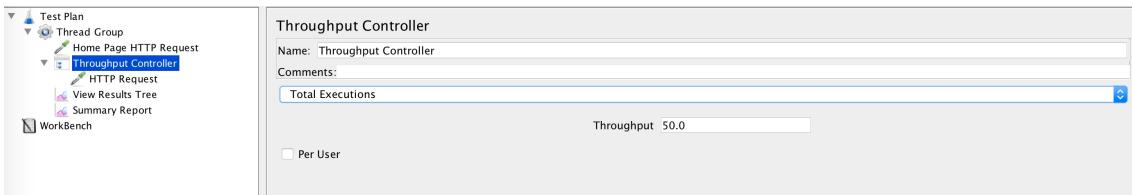
Throughput Controller

- 别被名字迷惑了，跟 吞吐量 没任何关系。用于控制多久执行一次，有两种方式: 百分比和 执行次数
- 应用场景: 限制请求的执行比率或执行次数
- 配制说明

- 执行类型

- Total Executions : 整个测试计划中总计执行次数
- Percent Executions : 整个测试计划中总计执行百分比
- Throughput : 设计的数值

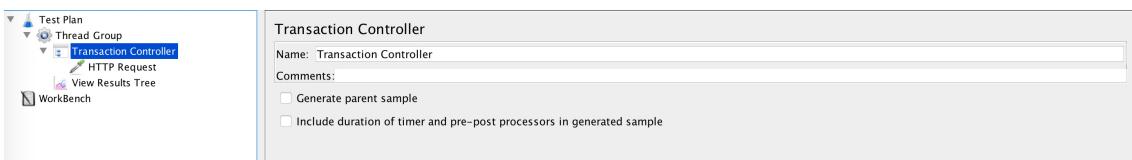
- `Per User` : 依据网上的说明在选择 `Total Executions` 时，勾选时会在每个线程中执行的次数。但在3.0版本中尝试使用无效



- 示例代码: [ThroughputController.jmx](#)

Transaction Controller

- 事务控制器，类似LR中的事务
- 应用场景: 完成一个完整的页面请求或一组请求
- 配制说明
 - `Generate parent sample` : 勾选后，所有的结果将在父结点中展示
 - `Include duration of timer and pre-post processors in generated sample` : 是否包括准备发送数据的CPU生成样例时间，默认不勾选



- 示例代码: [TransactionController.jmx](#)

While Controller

- 循环控制器，与开发语言中的 `While` 功能一致。直到条件为 `false` 时，停止运行
- 应用场景: 循环执行一个请求，仅判断一种状态下退出循环
- 配制说明

- `Condition(function or variable)` : 可配制为空/ `LAST` /变量或方法



- 示例代码: [WhileController.jmx](#)

断言测试

断言，对结果的判断。如：响应时间/返回状态码。

JMeter 丰富的断言支持

关于 Assertion 的使用，仅会挑选使用频率较高的进行详细说明。JMeter 自带了 12 种断言方式。



BeanShell Assertion



BSF Assertion



Compare Assertion



Duration Assertion



HTML Assertion



JSR223 Assertion



MD5Hex Assertion



Response Assertion



Size Assertion



SMIME Assertion



XML Assertion



XML Schema Assertion



XPath Assertion

- BeanShell Assertion
- BSF Assertion
- Compare Assertion
- Duration Assertion : 持续时间断言，主要用于响应时间的判断
- HTML Assertion
- JSR223 Assertion
- MD5Hex Assertion
- Response Assertion

- **Apply to** : 判断的范围选择，一般选择 `Main sample only`
 - `Main sample and sub-samples` : 主样例和子样例
 - `Main sample only` : 仅主样例
 - `Sub-samples only` : 仅子样例
 - `JMeter Variable` : JMeter 的变量
- **Response Field to Test** : 需要测试的内容，一般选择 `Text Response / Response Code` 和 `Response Header`
 - `Text Response` : 文件格式的返回内容，不包括返回的信息头
 - `Document(text)` : 文档
 - `URL Sampled` : URL样例
 - `Response Code` : 返回的状态码
 - `Response Message` : 返回信息
 - `Response Header` : 返回的信息头
 - `Ignore Status` : 忽略状态码
- **Pattern Matching Rules** : 匹配规则，一般选择 `Substring`
 - `Contains` : 包含
 - `Matches` : 匹配
 - `Equals` : 等于
 - `Substring` : 子字符串
 - `Not` : 没有
- **Patterns to Test** : 真正需要匹配的 测试内容，可通过下方的 `Add` `Delete` 按钮来维护。
- **Size Assertion** : 返回内容 大小 断言，可针对整个响应/响应头/响应数据体/响应信息进行判断
- **SMIME Assertion** : 主要用于邮件读取时的判断
- **XML Assertion** : 判断是否返回正确的 XML 格式内容
- **XML Schema Assertion** : 判断响应的内容与目标 XML 文件是否一致
- **XPath Assertion** : 判断返回内容中的 XPath 路径

使用 断言 实例

访问 <http://www.jianshu.com> 验证返回信息中是否包含 简书

- 正常访问 <http://www.jianshu.com> 查看是否在返回信息中存在 简书

```

<script type="text/javascript">...</script>
<script type="text/javascript">...</script>
<meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no">
<meta http-equiv="Cache-Control" content="no-siteapp">
<meta property="wl:webmaster" content="294ec9de89e7fadb">
<meta property="og:admin" content="104102651453316562112116375">
<meta property="dc:admin" content="11635613706305617">
<meta property="d:admin" content="11635616621163056375">
<meta name="google-site-verification" content="cv4-qkUJzR6gmFeajx_UyPe47QW9vY6cnCrYtCHYNh4">
<meta name="google-site-verification" content="HF7lff8YEGs1qtCE-kPml8Z469e2RHhGajy6JPVxI">
<meta http-equiv="mobile-agent" content="format=html5; url=http://www.jianshu.com/">
<title>首页 - 简书</title>
<meta name="csrf-param" content="authenticity_token">
<meta name="csrf-token" content="c9Fvh9ckX/th9sacEE6LXqrIvc77Xk6uIoLeXlz30nGX0uP+7SXluXHV2uYEDzJKcir28KwF0G67AL84E4N9Q==">
<!--[if lt IE 8]><script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script><![endif]-->
<link rel="stylesheet" media="all" href="http://cdn-qn0.jianshu.io/assets/base-28859e35d389885d08837bc971ff742c.css">
<link rel="stylesheet" media="all" href="http://cdn-qn0.jianshu.io/assets/reading-bc4422e046bfd9835fcdbb0c2de55d.css">
<link rel="stylesheet" media="all" href="http://cdn-qn0.jianshu.io/assets/base-read-mode-64accf6966299cfa3d580140a2582fe.css">
<script src="http://cdn-qn0.jianshu.io/assets/modernizr-613ea63b5aa2f0e2a1946e9c28c8eedb.js"></script>
<!-- Le HTML5 shim, for IE6-8 support of HTML elements -->
<!--[if lt IE 8]><link rel="stylesheet" media="all" href="http://cdn-qn0.jianshu.io/assets/scaffolding/for_ie-7f1c477ffedc13c11315103e8787dc6c.css" /><![endif]-->
<!--[if lt IE 9]><link rel="stylesheet" media="all" href="http://cdn-qn0.jianshu.io/assets/scaffolding/for_ie-7f1c477ffedc13c11315103e8787dc6c.css" /><![endif]-->
<link href="http://baijiji-common.b0.upaiyun.com/icons/favicon.ico" rel="icon">

```

- 使用 HTTP Request + Response Assertion + Result Tree

- 运行测试，查看结果，测试通过

Test Plan

- Thread Group
- HTTP Request
- Response Assertion
- View Results Tree**
- WorkBench

Name: View Results Tree

Comments:

Write results to file / Read from file

Filename:

Search: Case sensitive Regular exp. Search Reset

Text Sampler result Request Response data

```

<meta property="qc:admins" content="11635613706305617" />
<meta property="qc:admins" content="1163561616621163056375" />
<meta name="google-site-verification" content="c4-qkUjZR6gmFqajx_UyPe47GW9vY6cnCrYtCHYNh4" />
<meta name="google-site-verification" content="HF7fF8YEGrGsq1CE-kPml82469e2RhGajy6IPVY5Xl" />
<meta http-equiv="mobile-agent" content="format=html5; url=http://www.jianshu.com/">
```

<title>首页</title> **与真实的页面请求是一样的**

```

<meta name="csrf-param" content="JSESSIONID" />
<meta name="csrf-token" content="IYjbx0/USAJ2/C2BlipwEB7Cde027VuEfp1y8TjKa8NHFg6RTk2rk+1pjloMuHy9rWf4
Nge6Gq1V6y+1lg=" />
<!--[if lt IE 8]><script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script><![endif]-->
<link rel="stylesheet" media="all" href="http://cdn-qn0.jianshu.io/assets/base-28859e35d389885d08837bc971ff74
2c.css" />
```

```

<link rel="stylesheet" media="all" href="http://cdn-qn0.jianshu.io/assets/reading-bc4422e046bfdb9835cfccdb0c2
de55d.css" />
<link rel="stylesheet" media="all" href="http://cdn-qn0.jianshu.io/assets/base-read-mode-64accf6966299cfa3d58
0140a2582fe.css" />
<script src="http://cdn-qn0.jianshu.io/assets/modernizr-613ea63b5aa2f0e2a1946e9c28c8edb.js"></script>
<!-- Le HTML5 shim, for IE6-8 support of HTML elements -->
<!--[if lt IE 9]><link rel="stylesheet" media="all" href="http://cdn-qn0.jianshu.io/assets/scaffolding/for_ie_7f1c477f
edc13c11315103e8787dc6.css" /><![endif]-->
<!--[if lt IE 9]><link rel="stylesheet" media="all" href="http://cdn-qn0.jianshu.io/assets/scaffolding/for_ie_7f1c477
ffedc13c11315103e8787dc6.css" /><![endif]-->
<link href="http://baiji-common.b0.upaiyun.com/icons/favicon.ico" rel="icon">
<link rel="apple-touch-icon-precomposed" href="http://cdn-qn0.jianshu.io/assets/apple-touch-icons/57-b426
758a1fcfb30486f20fd073c3b8ec.png" sizes="57x57" />
<link rel="apple-touch-icon-precomposed" href="http://cdn-qn0.jianshu.io/assets/apple-touch-icons/72-feca4
b183b929fd188665785dc7a7f1.png" sizes="72x72" />
<link rel="apple-touch-icon-precomposed" href="http://cdn-qn0.jianshu.io/assets/apple-touch-icons/76-ba757
f1ad3421192ce7192170393d2b0.png" sizes="76x76" />
<link rel="apple-touch-icon-precomposed" href="http://cdn-qn0.jianshu.io/assets/apple-touch-icons/114-8dae
200.css" />
```

Find Case sensitive Regular exp.

Scroll automatically?

追加对返回 状态码 的验证，提高测试的准确性

- 再添加一个新的 Response Assertion

Test Plan

- Thread Group
- HTTP Request
- Response Content Assertion
- Response Code Assertion**
- View Results Tree
- WorkBench

Name: Response Assertion

Comments:

Apply to:

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

Response Field to Test

Text Response Document (text) URL Sampled Response Code Response Message Response Headers Ignore Status

Pattern Matching Rules

Contains Matches Equals Substring Not

Patterns to Test

200	状态码
-----	-----

- 执行测试，查看测试结果

The screenshot shows the JMeter interface with the 'View Results Tree' listener selected. The left sidebar displays a test plan with a single thread group containing an HTTP request. The main window shows the results of that request.

Test Plan Structure:

- Test Plan
 - Thread Group
 - HTTP Request

View Results Tree Listener Configuration:

- Name: View Results Tree
- Comments: Write results to file / Read from file
- Filename: (empty)
- Log/Display Only: Errors, Successes (unchecked)
- Configure button

Request Details:

- Thread Name: Thread Group 1-1
- Sample Start: 2016-09-02 13:15:14 CST
- Load time: 275
- Connect Time: 82
- Latency: 208
- Size in bytes: 37989
- Headers size in bytes: 1137
- Body size in bytes: 36852
- Sample Count: 1
- Error Count: 0
- Data type ("text"|"bin"): text
- Response code: 200 (highlighted with a blue box)
- Response message: OK

Response Headers:

- Response headers:
HTTP/1.1 200 OK
Server: nginx
Date: Fri, 02 Sep 2016 05:09:38 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 36852
Connection: keep-alive
Vary: Accept-Encoding
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
ETag: W/"df1453feeb7de717da8ef7131c64842"
Cache-Control: max-age=0, private, must-revalidate
Set-Cookie: signin_redirect=http%3A%2Fwww.jianshu.com%2F; path=/
Set-Cookie: read_mode=day; path=/
Set-Cookie: default_font=font2; path=/
Set-Cookie: f Ankia=

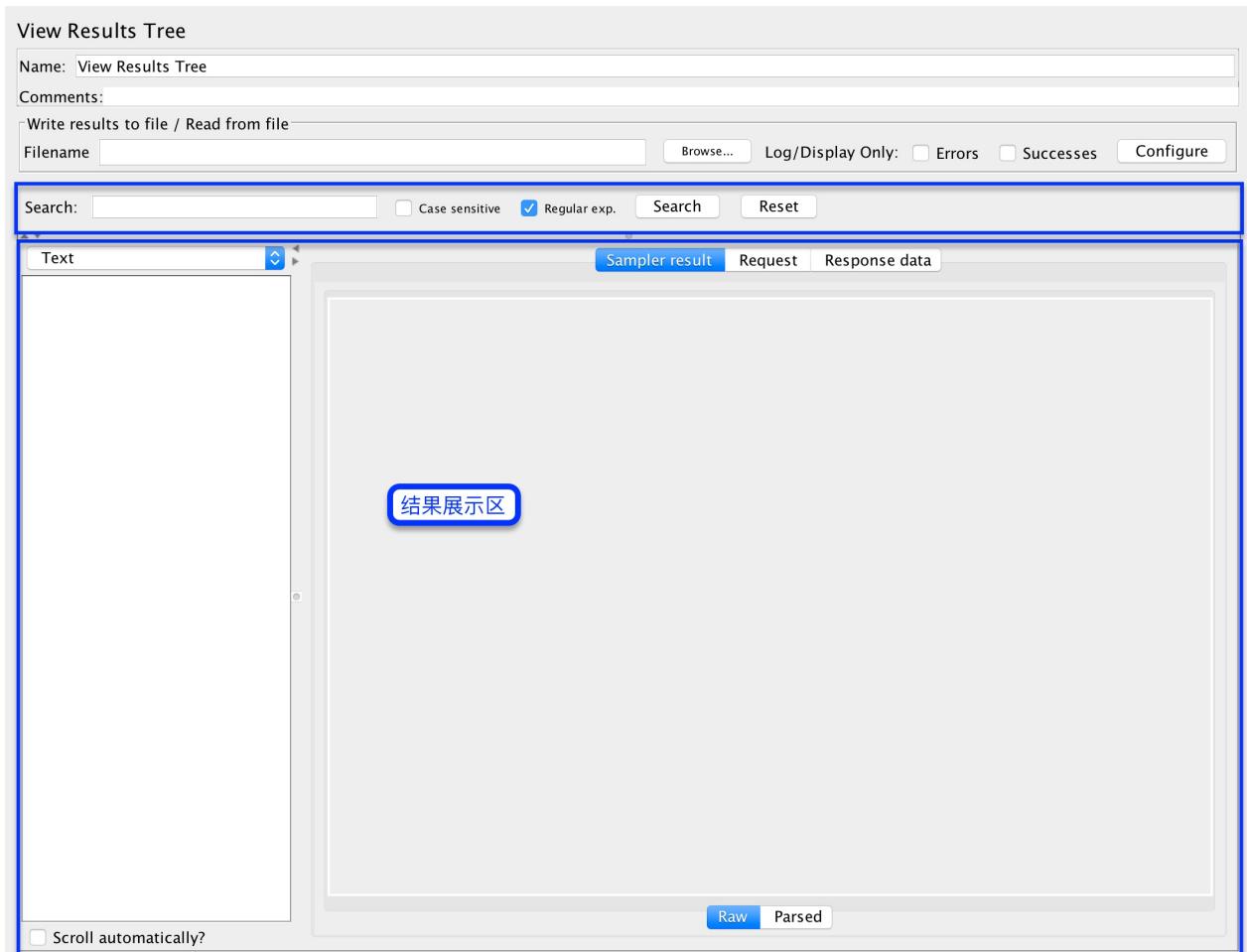
Buttons at the bottom:

- Raw
- Parsed (selected)

A blue callout box points to the 'Response code: 200' entry in the results table, with the text "通过JMeter请求，返回的状态码" (Status code returned by the JMeter request) inside it.

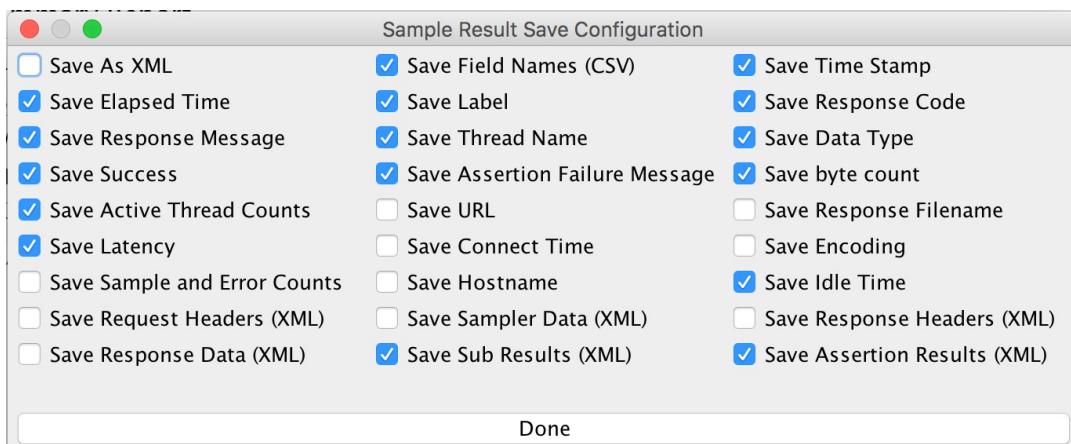
Result Tree

主要用于结果的测试执行结果的展示，会展示样例的 结果 / 请求信息 / 返回信息 。



配制说明

- `Write results to file / Read from file` : 写结果到文件或读取文件内容
 - `File` : 需要加载的文件
 - `Log/Display Only`
 - `Errors` : 仅显示错误
 - `Successes` : 仅显示成功的部分
 - `Configure` : 需要配制保存的字段



- Search : 快速搜索样例名

- Case sensitive : 启用大小写敏感搜索
- Regular exp. : 启用正则表达式搜索
- Search Reset : 搜索/重置按钮

- 结果展示区

- Text : 文件样式查看样例结果，可通过下方的 Raw Parsed 来切换展示方式。
- RegExp Tester : 正则表达式针对返回结果的调试器
- CSS/JQuery Tester : CSS/JQuery 的调试器
- XPath Tester : XPath 调试器
- HTML : HTML 文件形式查看，不会下载任何页面中的资源
- HTML(download resources) : 下载页面资源，再以 HTML 样式展示，但样式不会使用
- Document : 主要用于有外部文件时，展示页面的结果。如:Word/Excel文件。
- JSON : 以 JSON 数据格式展示
- Scroll automatically? : 勾选后，在测试过程中会自动滚动结果树中的展示，以及时展示最新的测试结果

应用

- 调试器 可用于在返回内容中提取 变量，如: 用户名
- 在性能测试时， 结果树 一定要取消，因为它会占用大量的资源(内存和CPU)。

Summary Report

主要用于 实时 测试结果的汇总查看，如: 请求时间/错误率/吞吐量/平均请求字节数

配制说明

Summary Report

Name:	Summary Report																																																			
Comments:																																																				
Write results to file / Read from file																																																				
Filename	<input type="button" value="Browse..."/>	<input type="checkbox"/> Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes <input type="button" value="Configure"/>																																																		
<table border="1"> <thead> <tr> <th>Label</th> <th># Samples</th> <th>Average</th> <th>Min</th> <th>Max</th> <th>Std. Dev.</th> <th>Error %</th> <th>Throughput</th> <th>KB/sec</th> <th>Avg. Bytes</th> </tr> </thead> <tbody> <tr> <td>HTTP Reque...</td> <td>1000</td> <td>418</td> <td>163</td> <td>1412</td> <td>152.61</td> <td>0.00%</td> <td>33.4/sec</td> <td>134.96</td> <td>4134.1</td> </tr> <tr> <td>HTTP Reque...</td> <td>1000</td> <td>397</td> <td>170</td> <td>796</td> <td>128.89</td> <td>0.00%</td> <td>33.7/sec</td> <td>387.54</td> <td>11767.0</td> </tr> <tr> <td>HTTP Reque...</td> <td>1000</td> <td>396</td> <td>165</td> <td>587</td> <td>129.67</td> <td>0.00%</td> <td>33.8/sec</td> <td>173.57</td> <td>5260.0</td> </tr> <tr> <td>TOTAL</td> <td>3000</td> <td>404</td> <td>163</td> <td>1412</td> <td>137.88</td> <td>0.00%</td> <td>97.7/sec</td> <td>672.91</td> <td>7053.7</td> </tr> </tbody> </table>			Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes	HTTP Reque...	1000	418	163	1412	152.61	0.00%	33.4/sec	134.96	4134.1	HTTP Reque...	1000	397	170	796	128.89	0.00%	33.7/sec	387.54	11767.0	HTTP Reque...	1000	396	165	587	129.67	0.00%	33.8/sec	173.57	5260.0	TOTAL	3000	404	163	1412	137.88	0.00%	97.7/sec	672.91	7053.7
Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes																																											
HTTP Reque...	1000	418	163	1412	152.61	0.00%	33.4/sec	134.96	4134.1																																											
HTTP Reque...	1000	397	170	796	128.89	0.00%	33.7/sec	387.54	11767.0																																											
HTTP Reque...	1000	396	165	587	129.67	0.00%	33.8/sec	173.57	5260.0																																											
TOTAL	3000	404	163	1412	137.88	0.00%	97.7/sec	672.91	7053.7																																											
<input type="checkbox"/> Include group name in label? <input type="button" value="Save Table Data"/> <input checked="" type="checkbox"/> Save Table Header																																																				

- Write results to file / Read from file

- **Filename** : 选择需要加载或保存测试结果的文件

- **Log/Display Only**

- **Errors** : 仅显示有错误的结果

- **Successes** : 仅显示成功的结果

- **Configure** : 配制需要保存的字段数据

Sample Result Save Configuration

<input type="checkbox"/> Save As XML	<input checked="" type="checkbox"/> Save Field Names (CSV)	<input checked="" type="checkbox"/> Save Time Stamp
<input checked="" type="checkbox"/> Save Elapsed Time	<input checked="" type="checkbox"/> Save Label	<input checked="" type="checkbox"/> Save Response Code
<input checked="" type="checkbox"/> Save Response Message	<input checked="" type="checkbox"/> Save Thread Name	<input checked="" type="checkbox"/> Save Data Type
<input checked="" type="checkbox"/> Save Success	<input checked="" type="checkbox"/> Save Assertion Failure Message	<input checked="" type="checkbox"/> Save byte count
<input checked="" type="checkbox"/> Save Active Thread Counts	<input type="checkbox"/> Save URL	<input type="checkbox"/> Save Response Filename
<input checked="" type="checkbox"/> Save Latency	<input type="checkbox"/> Save Connect Time	<input type="checkbox"/> Save Encoding
<input type="checkbox"/> Save Sample and Error Counts	<input type="checkbox"/> Save Hostname	<input checked="" type="checkbox"/> Save Idle Time
<input type="checkbox"/> Save Request Headers (XML)	<input type="checkbox"/> Save Sampler Data (XML)	<input type="checkbox"/> Save Response Headers (XML)
<input type="checkbox"/> Save Response Data (XML)	<input checked="" type="checkbox"/> Save Sub Results (XML)	<input checked="" type="checkbox"/> Save Assertion Results (XML)

Done

- 展示的结果区域

- **Label** : 展示测试过程中所有执行的请求和控制器的 聚合

- **# Samples** : 样例执行数量

- **Average** : 平均时间, 单位: ms

- **Min** : 最小时间, 单位: ms

- **Max** : 最大时间, 单位: ms

- **Std. Dev.** : Standard deviation 标准的偏移量

- **Error %** : 错误率

- **Throughput** : 服务每秒处理的请求数量

- **KB/sec** : 自身的吞吐量

- **Avg. Bytes** : 响应的平均大小

- **Include group name in label** : 在 Label 中添加组名显示

- **Save Table Data** : 保存 结果区域 数据至 CSV

- Save Table Header : 保存数据时，是否保存表头。如: Label/# Samples

The screenshot shows the 'Summary Report' configuration dialog in JMeter. On the left, there is a tree view of the test plan: 'Test Plan' > 'Thread Group' > 'HTTP Request' > 'Summary Report'. The 'Summary Report' node is selected. The main area is titled 'Summary Report' and contains fields for 'Name: Summary Report' and 'Comments:'. Below these are options for 'Write results to file / Read from file' and a 'Filename' input field. To the right are buttons for 'Browse...', 'Log/Display Only:', 'Errors', 'Successes', and 'Configure'. A table is displayed showing performance metrics:

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes
Thread Group-HTTP Request	10	314	212	669	172.85	0.00%	2.6/sec	96.91	38216.4
TOTAL	10	314	212	669	172.85	0.00%	2.6/sec	96.91	38216.4

At the bottom, there are checkboxes for 'Include group name in label?' (unchecked), 'Save Table Data' (unchecked), and 'Save Table Header' (checked). There is also a 'Configure' button.

应用

- 测试整体计划的调试: 执行次序/错误率/吞吐量
- 测试结果的保存与展示

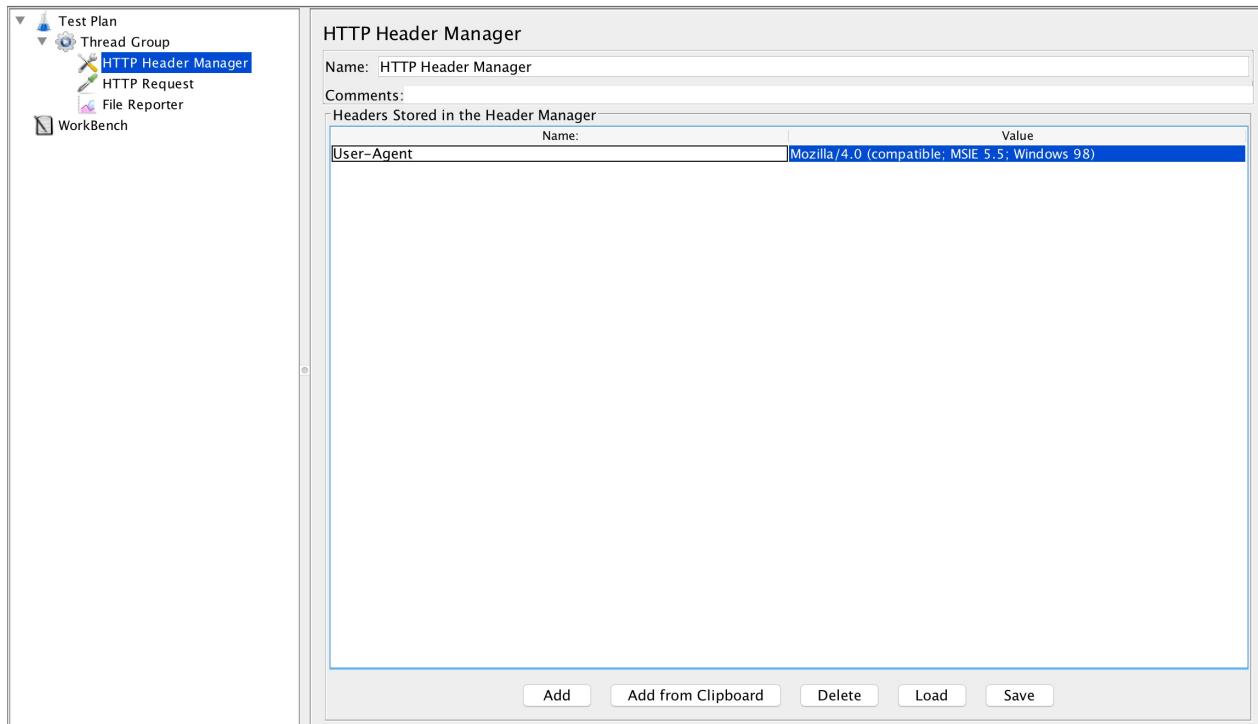
完善JMeter

- [HTTP信息头管理](#)
 - `HTTP Header Manager`，主要用于设计 `HTTP` 消息头设置
 - `HTTP Cookie Manager`，主要用于设计 `HTTP` 请求时的Cookie配制
- [数据获取](#)
 - 使用 `CSV` 文件来获取数据，实例 `CSV`
 - 用户自定义变量，数据使用
 - 正则表达式，从响应数据中提取变量
- [实战性能测试](#)
 - 使用[JMeter](#)进行一次完整项目实践

HTTP信息头管理

用于 添加 重写 HTTP请求的头信息。如： User-Agent Accept-Encoding

配制说明



- Headers Stored in the Header Manager : 用于配制 HTTP 请求时的头信息
 - Name : 头信息的 属性名
 - Value : 头信息的 属性值
- 示例代码: [HeaderManager.jmx](#)

应用

```

▼ Request Headers view source
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8,zh-CN;q=0.6,zh;q=0.4,zh-TW;q=0.2
Cache-Control: no-cache
Connection: keep-alive
Cookie: default_font=font2; Hm_lvt_0c0e9d9b1e7d617b3e6842e85b9fb068=1468838203,1468906047,1468991137,1469172578; gr_user_id=a0b2fc80-7c2a-4ab4-9326-de2773e7beb2; read_mode=day; _session_id=Tz1Jem9XbmtuGZHMdVVWJaSTRSVnVVT1NuTtKNSCv1TE91VjZsVU9yMmIrxKA0NjVwempGOU8vT3qwd3RjczJ1U2xYdUgxZEphLUk<4Q1N3K0U3Un0d5ERKSHE50FJ4NWLRaStob1d5WY5b0dLTDA0c0ZyTc6WtONnhubF25c026THFDYXk27Hg2RnBjbnovRHk0MkFkew5XwWe1ehlyQUrVGzsbdVM3U5em1tb6JLVL1ltanFHU1VRabhr0XV2Wnd0ThhYUjh6a2RCWxZoNnj1MN10TMwaENXNnjJzR1Rur09wbXNDRVJYemszdUM5UmE0TkvTeW16509MOVlVsTVZJaXkXUm4zD2tVxzRw450223MkE1U3OYXc0eDwRkcvc1l1I0DQ2NzBLau1XevVRdvk5UmkyTm1R0GtBanduVSlaGM1WhVvYUlxLydzdXXbRk2FPNs1UanJnNdTNl9VYnjXg5pdllnwWfHVfpbHZRUzF5kdMT1jNMV5BZHRwZjZjd28WlitvTNWw1ElytHVE1V2Opid3RNWSmtdThVtk1hNvZPb59sYUduR2HMCUvnQkvNuzM0R8pRTlpYnxF4emh6SHFZ0XU2a1FaQuNvOFJPcFLdb3drXpTVGFxRfNm50U9LS1LTWpbzbE4h0tWQwp1QmZqYXBmbG9RPT0t607c02ad87d23f/bbbb62f961ct955; CNZZDATA1258679142=1279397371-1471409633-471473049431; __utmt=1; __utma=194070582,1473673783,1458978711,1473036398,1473050824,127; __utmb=194070582,1.10.1473050824; __utmc=194070582; __utmc=194070582,1472432425,117,23,utmcsr=aimer1124.github.io|utmccn=(referral)|utmcmd=referral|utmccct=2016/08/14/Book-%E6%97%A0%E5%A3%80%E5%91%8A%E7%99%BD/; __utmv=194070582,2>User%20Type=Member=1
Host: www.jianshu.com
Pragma: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/52.0.2743.116 Safari/537.36

```

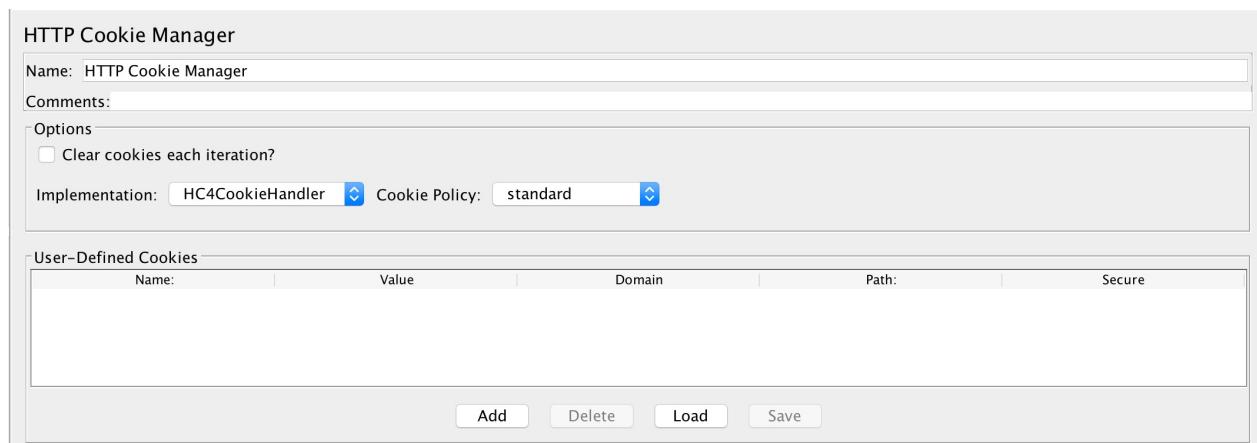
以上 Request Headers 中的内容均可使用 HTTP Header Manager 来模拟，下面简要说明部分使用。

- 使用 User-Agent 来模拟不同的客户端发起的请求，如：手机端/指定浏览器版本
- 使用 Accept 来模拟文件格式
- 使用 Cookie 中的配制来模拟发送请求中的 Cookie 信息，使用 HTTP Cookie Manager 可进行较为复杂的 Cookie 管理和维护

HTTP Cookie 管理器

- 可以像浏览器一样 存储 发送 Cookies。
- 手动管理和维护 Cookies 中的内容。

配制说明



- Options
 - Clear cookies each iteration? : 勾选后，每次 线程组 的循环均会清空 Cookie，但 自定义 的不会被清空
 - Implementation : 选择需要继承的HC4CookieHandler(HttpClient 4.5.X API)/HC3CookieHandler(HttpClient 3 API)，可使用默认
 - Cookie Policy : 先找 Implementation 的具体策略，可使用默认
- User-Defined Cookies : 用户自定义Cookie
 - Name : Cookie的 属性名
 - Value : Cookie的 属性值
 - Domain : 作用域，对于有 作用域 要求的
 - Path : 域路径
 - Secure : 安全性，可使用默认不勾选
- 示例代码: [CookieManager.jmx](#)

应用

- 业务需求需要在每次执行时清空 `Cookie` 时，可勾选 `Clear cookies each iteration?`
- 指定 `Cookie` 内容，可在 `User-Defined Cookies` 添加自定义的 `Cookie`

JMeter 使用过程中，会需要引入 外部数据 / 过程数据 / 结果数据 。以下三种技术方案将帮你实现此需求。

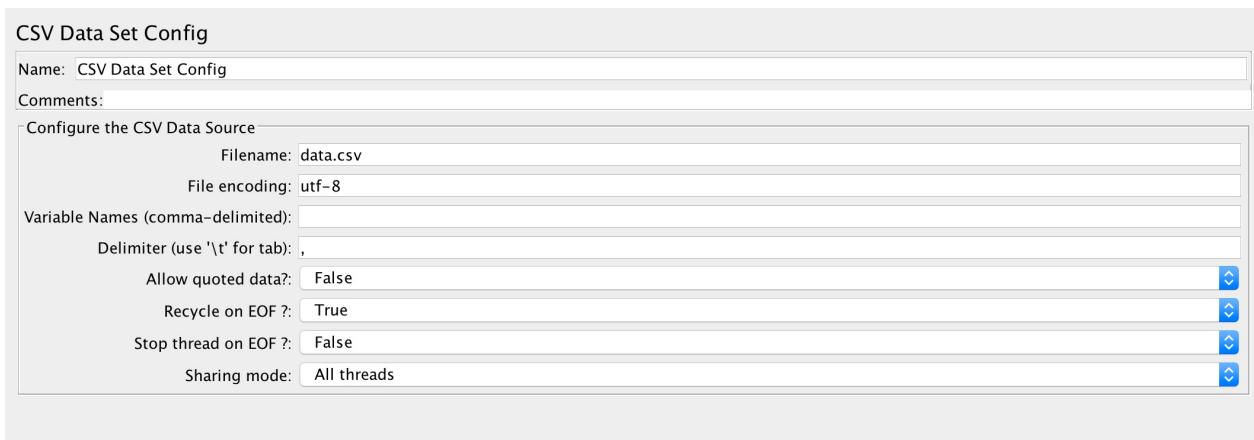
CSV Data Set Config

用于配制 csv 文件数据集。

CSV 文件

- CSV 全称: Comma-Separated Values ,逗号分隔值或字符分隔值。其文件以纯文本形式存储表格数据（数字和文本）。
- 详细的 csv 文件说明可参考 [百度文库-CSV](#)

配制说明



- Config the CSV Data Source
 - Filename : 获取目标的 csv 文件，建议与测试计划jmx文件存放在同和文件夹中
 - File encoding : 文件编码格式，有需求时，可配制。默认留空即可
 - Variable Names(comma-delimited) : 定义变量名，以逗号分隔不同的变量名。若此外不定义，则会取 csv 文件中的首行内容为变量名。建议此处不定义
 - Delimiter(use '\t' for tab) : 使用哪种符号来分隔，默认使用 ,
 - Allow quoted data? : 如果设置为 True ，则在变量中可以使用 双引号 。默认为 False
 - Recycle on EOF? : 在线程组运行过程中，若文件读取至最后一行后，是否 继续 读取。默认为 True
 - Stop thread on EOF? : 在线程组运行过程中，若文件读取至最后一行后，是否 停止 线程。默认为 False

- Sharing mode : 读取数据的分享模式
- All threads : 文件数据在所有线程中共享
- Current thread group : 仅在单个 线程组 中共享
- Current thread : 仅针对单个 线程 共享
- 示例参考: 仅用于说明 csv 文件调用的使用, 可在 Result Tree 中的 Request 中查看到
 - 测试计划: [CSVDataSetConfig.jmx](#)
 - csv 文件: [data.csv](#)

应用

- 并发测试中针对不同用户信息进行获取, 只需要下面**3步**。
 - 先将需要测试的用户账号信息存入至 csv 文件中
 - 在 HTTP 请求信息中将账号信息 参数化
 - 使用[JMeter](#)的 CSV Data Set Config 元素来调用 csv 文件, 并将数据传递给 HTTP 请求
- 一些初始化数据的使用, 如: 特定的 定单 状态

User Defined Variables

使用性能测试时, 会需求一些提前准备的数据。如: 指定的用户名/指定的订单号。使用 User Defined Variables 可实现。

配制说明

User Defined Variables

Name:	User Defined Variables	
Comments:		
User Defined Variables		
Name:	Value	Description
username	zhangsan	用户名

[Detail](#) [Add](#) [Add from Clipboard](#) [Delete](#) [Up](#) [Down](#)

- User Defined Variables : 存放所有的用户自定义变量数据
 - Name : 变量名
 - Value : 变量值
 - Description : 变量的描述信息
- Detail : 详情功能，点击后可展开并修改 Variable 的具体信息
- Add : 添加一条新的变量，默认内容均为空
- Add from Clipboard : 将当前复制的内容添加至 Variable 中
- Delete : 删除一条 Variable 记录
- Up/Down : 上移/下移变量的位置
- 示例参考: [UserDefinedVariables.jmx](#)

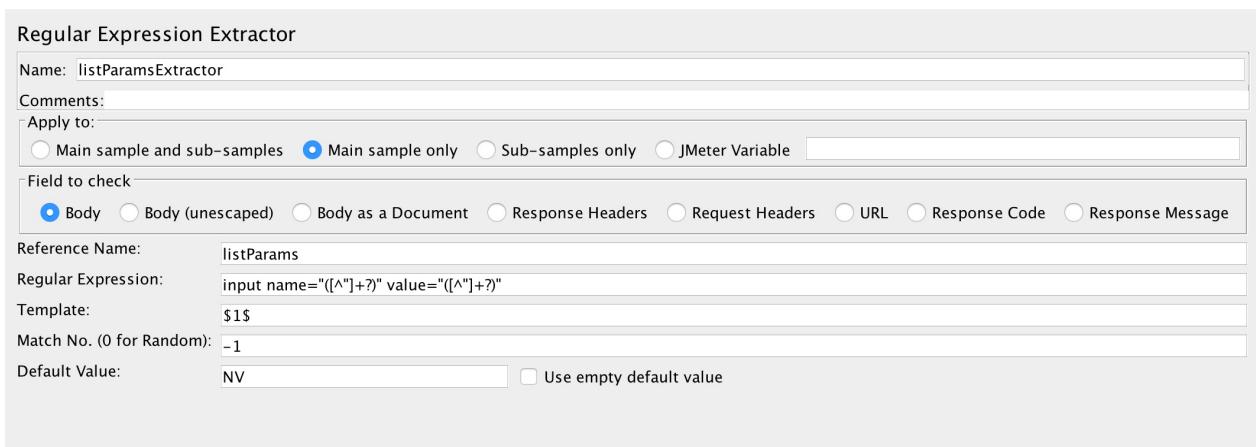
应用

- 自定义变量设置: 全局变量时，可使用此功能。如: 用户ID/订单号

Regular Expression Extractor

使用 正则表达式 提取 数据，并在 JMeter 中使用

配制说明



- **Apply to** : 需要匹配的样例应用范围
 - Main sample and sub-samples : 主样例和子样例
 - Main sample only : 主样例
 - Sub-samples only : 仅子样例
 - JMeter Variable : JMeter 变量
- **Field to check** : 检查的区域
 - Body : 返回的数据体，如：页面的内容(除消息头)。默认使用
 - Body(unescaped) : 所有的 HTML 均被替换。慎用
 - Body as a Document : 像文档的数据体
 - Response Headers : 返回的消息头
 - Request Headers : 请求的消息头
 - URL : URL 请求信息，如：www.baidu.com
 - Response Code : 返回的状态码
 - Response Message : 返回信息
- **Reference Name** : 抽取出来的变量名
- **Regular Expression** : 匹配的 正则表达式
- **Template** : 匹配模板，从匹配结果中创建一组字符串。若 Regular Expression 中会抽取多个结果，则 \$1\$ 是第一组数据/ \$2\$ 表示第二组数据。 \$0\$ 表示整个结果
- **Match No.(0 for Random)** : 使用匹配的第 No 结果，0 表示随机，负数用于与 ForEach Controller 配合使用
- **Default Value** : 默认值，如果没有匹配出结果，则使用的默认值。
 - Use empty default value : 勾选后，则使用 空 为默认值
- 示例代码: [RegularExpressionExtractor.jmx](#)

应用

- 对于返回结果中数据提取，如：从第一个请求中获取 用户信息，再从第二个请求中使用 用户信息
- 配合 `ForEach Controller` 进行对应结果的场景处理，如：从第一个请求中获取一组数据，再依次结合这一组数据获取另外的数据

实战性能测试

目的

- 学习使用 JMeter 工具，实现针对 Web 项目 URL 的并发请求，完对结果进行整理及分析。
- 找出性能测试的 三步曲

使用技术点

清单

- `HTTP Request` : 发送 HTTP 请求
- `ForEach Controller` : 循环处理
- `Response Assertion` : Response 断言
- `Regular Expression Extractor` : 正则表达式提取器
- `Debug Sampler` : 调试样例
- `RegExp Tester` : 正则表达式测试
- `View Results Tree` : 查看结果树
- `Assertion Results` : 断言结果
- `Summary Report` : 概要报告

实例

需求：查看简书首页所有文章链接的访问时间

拆分

- 只有实时通过获取 简书首页 内容，才能获取 首页 中 所有文件章 链接
- 通过使用 文章 链接的访问，才能达到测试访问时间的目的
- 对获取 文章页 的状态码进行校验

调试

Step1: 获取文章内容

- 添加 `HTTP Request`，来获取 简书 首页 内容



- 添加 Result Tree，来查看运行结果。执行测试，查看结果：可正常请求，并返回状态码

The screenshot shows the JMeter interface with a 'View Results Tree' listener selected. The results pane displays a single sample named '获取“简书”首页'. The response code is highlighted as 200. The response data pane shows the full HTML response from the Jianshu homepage, including headers and the body content.

Step2: 对获取的内容进行分析，找到需要匹配的部分

- 使用 Result Tree 中的结果，分析返回数据格式
 - 返回的所有文章均在 `<ul class="article-list thumbnails">` 中，且每个文章均被存放在元素 `li` 中

```

<li class=have-img>
    <a class="wrap-img" href="/p/c5f3bd6836a9"></a>
    <div>
        <p class="list-top">
            <a class="author-name blue-link" target="_blank" href="/users/1441f4ae075d">彭小六</a>
            <em>·</em>
            <span class="time" data-shared-at="2016-09-07T11:28:34+08:00"></span>
        </p>
        <h4 class="title"><a target="_blank" href="/p/c5f3bd6836a9">5万粉丝、月入6位数，1年时间简书带给了我什么</a></h4>
        <div class="list-footer">
            <a target="_blank" href="/p/c5f3bd6836a9">
                阅读 3822
            </a>      <a target="_blank" href="/p/c5f3bd6836a9#comments">
                · 评论 49
            </a>      <span> · 喜欢 162</span>
            <span> · 打赏 2</span>
        </div>
    </div>
</li>

```

- 文章的标题及链接被存放在 `<h4 class="title">5万粉丝、月入6位数，1年时间简书带给了我什么</h4>` 中
- `a` 标签中的 `href` 值: `/p/c5f3bd6836a9`
- 访问可正常打开文章 `5万粉丝、月入6位数，1年时间简书带给了我什么` 的内容
- 只需要将所有文章元素中的 `href` 提取出来，再将 `www.jianshu.com` 的添加上，便可完成对所有文章请求的链接拼装

Step3: 对返回内容结果进行调试，找出正则表达式

- 使用 `Result Tree` 中的 `RegExp Tester` 调试，最终达到提取文章 `href` 内容。

- 最终的正则表达式为

The screenshot shows the JMeter interface with the 'View Results Tree' listener selected. The results pane displays an HTML page with various elements. A blue box highlights the 'Regular expression' input field containing the regex: <h4 class="title">. The 'Test' button next to it is also highlighted. Below the input field, the 'Match count' is shown as 20, followed by a list of 20 matches. The entire results pane is framed by a blue border.

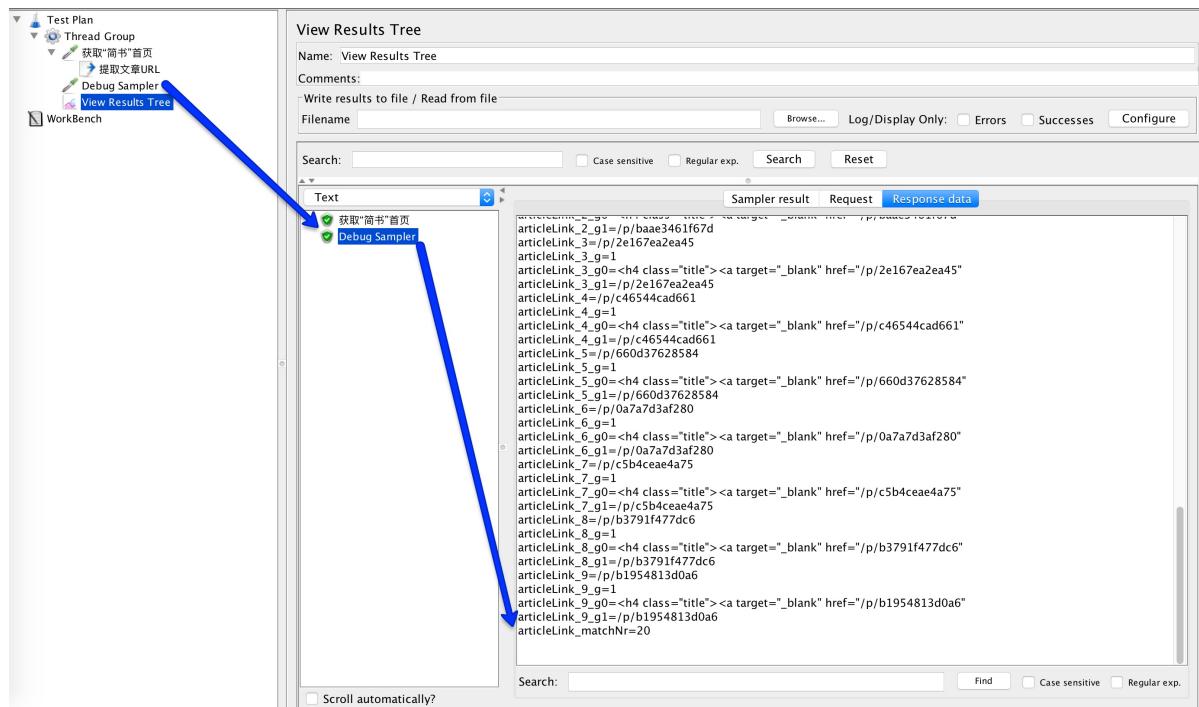
```
<h4 class="title"><a target="_blank" href="(.*?)>
```

Step4: 添加正则表达式提取器，提取获取内容

- 添加 Regular Expression Extractor 来提取首页中获取的每个文章的 href 值，并返回给变量 articleLink

The screenshot shows the JMeter interface with the 'Regular Expression Extractor' configuration dialog open. The 'Name' field is set to '提取文章URL'. The 'Regular Expression' field contains the same regex as above: <h4 class="title">. The 'Template' field is set to '\$1\$'. The 'Default Value' field is empty. The 'Field to check' section is set to 'Body'. The 'Reference Name' is 'articleLink'. The 'Apply to' section has 'Main sample only' checked. The entire configuration dialog is framed by a blue border.

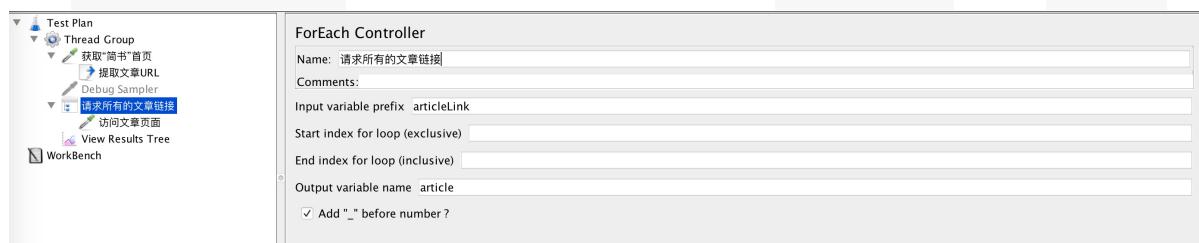
- 添加 Debug Sampler 查看 正则表达式 提取的结果是否正确，使用默认配制即可
- 再次运行测试，查看 Result Tree 中查看 Debug Sampler 的结果



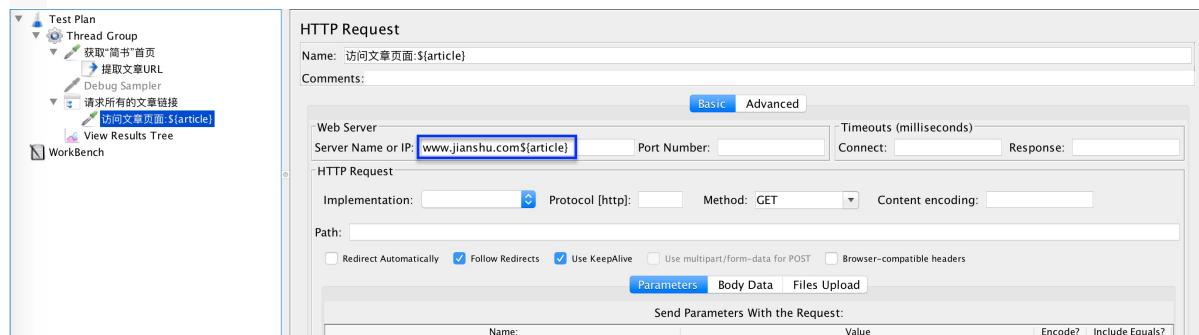
- 至此，提取文章 href 值的内容已经完成，禁用 Debug Sampler

Step5: 获取所有文章内容

- 使用 ForEach Controller 完成对获取内容进行处理。输入 articleLink ，返回 article



- 在 ForEach Controller 内，添加请求文章页面的 HTTP Request，并将变量 article 拼装至请求中



- 再次执行测试，查看 Result Tree 中的测试结果

The screenshot shows the JMeter interface with the 'View Results Tree' listener selected in the left sidebar. The main pane displays a tree view of requests under the category '访问“简书”首页'. Each request is shown with a green checkmark and a link to its details. The right pane shows the raw response content for one of the requests, which is a large block of HTML and JavaScript code. A blue arrow points from the text above to the 'View Results Tree' button in the sidebar.

Step6 添加对访问文章页面正确性的验证：状态码 200

- 在请求文章内容页面内，添加 Response Assertion，校验返回的状态码是否为 200

The screenshot shows the JMeter interface with the 'Response Assertion' configuration dialog open. The 'Name' field is set to '响应状态码判断-200'. The 'Apply to:' dropdown is set to 'Main sample and sub-samples'. The 'Response Field to Test' dropdown is set to 'Response Code'. The 'Patterns to Test' section contains the value '200', with a blue arrow pointing to it. Other options like 'Contains', 'Matches', 'Equals', 'Substring', and 'Not' are also visible.

- 添加 Assertion Results 查看所有的断言结果，执行测试查看结果

The screenshot shows the JMeter Test Plan interface. On the left, under the Thread Group, there is a 'Assertion Results' sampler. On the right, the 'Assertion Results' panel is open, showing the name 'Assertion Results' and a list of URLs under the 'Assertions:' section. The URLs listed are all variations of '访问文章页面:/p/[long_hexadecimal_string]'.

至此，所有的调试工作已完成

测试

优先JMeter的配制

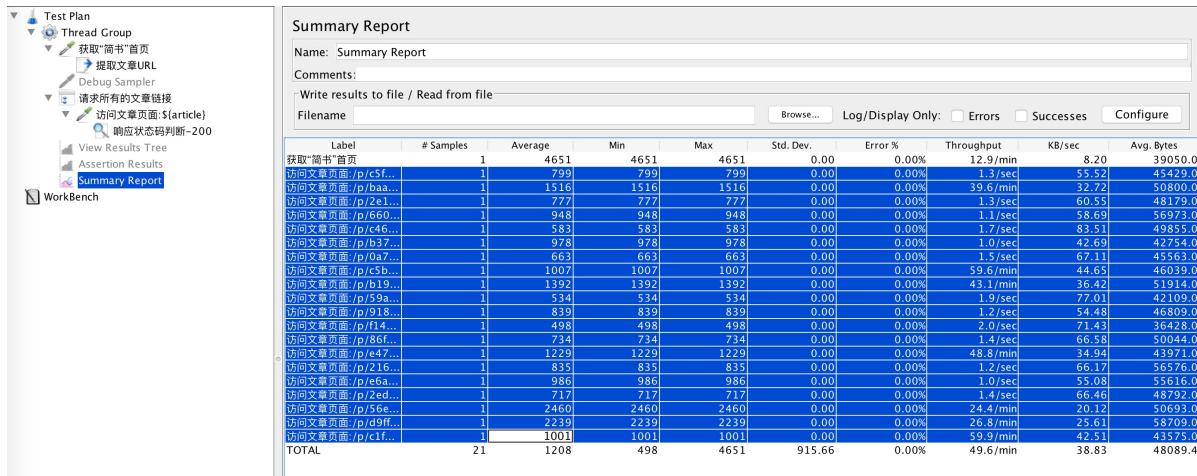
- 禁用 Assertion Results
- 禁用 Result Tree
- 添加 Summary Report ，查看测试概要结果

The screenshot shows the JMeter Test Plan interface with a 'Summary Report' sampler added. On the right, the 'Summary Report' panel is open, showing the name 'Summary Report' and a table of results. The table has one row labeled 'TOTAL' with the following data:

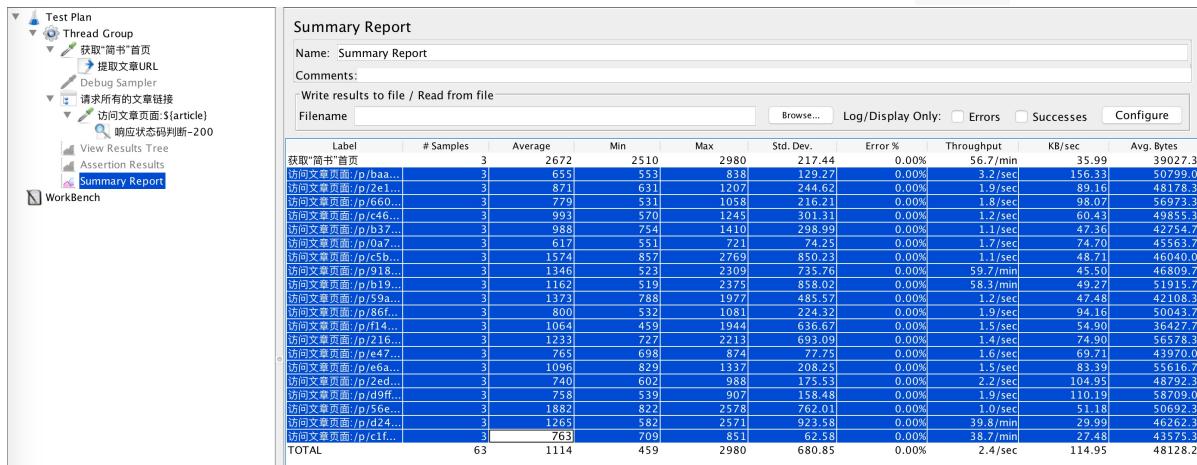
Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes
TOTAL	0	0	92233720368...	-9223372036...	0.00	0.00%	.0/hour	0.00	.0

执行压力测试

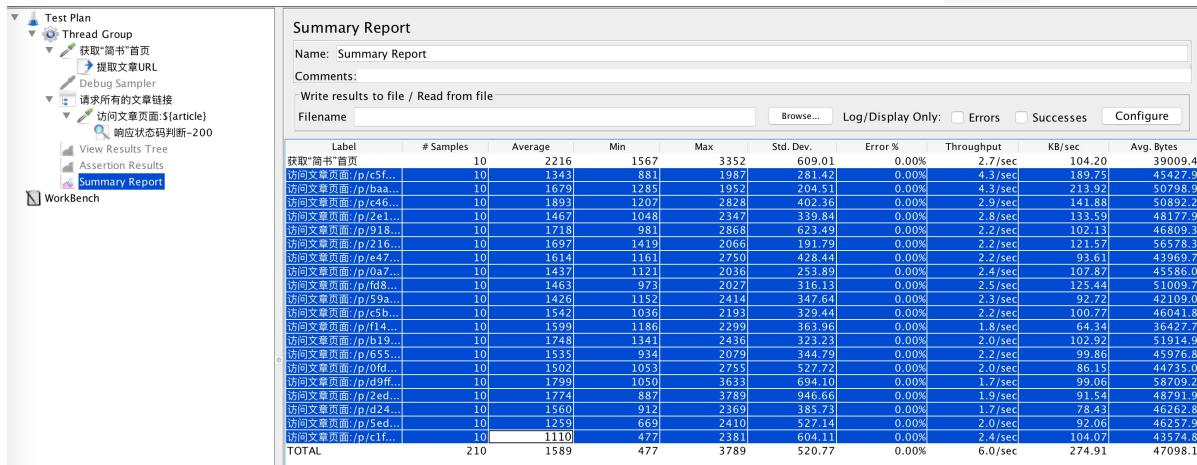
- 设置 Thread Group 中的 Number of Thread(users) 为1，并发为1，查看测试结果: 最小的498至最大的2460，单位: 毫秒。错误率为0



- 调整并发为3时，查看测试结果：最小655至最大1574，单位：毫秒。错误率为0



- 调整并发为10时，查看测试结果：最小1110至最大1893，单位：毫秒。错误率为0



-
-
- 可以逐步将并发数据提升，以达到最终找出系统的性能阈值。性能测试结束

总结

- 完整示例代码：[Sample.jmx](#)
- 性能测试三步走：分析业务点->调试脚本->上测试，上面已经完整展示。

- 通过上面几组的测试数据，可以看出，第一次的最大值 2460，在后面的测试过程中再也没有出现过。有可能是网络波动引起的
- 此项目访问的正式营运网站简书，因此不能直接将并发量提高太多，不能出现恶意访问的情况，练手即可。别被封IP或被告
- 不足
 - 当前设计的测试计划在执行过程中会有 干扰 数据：访问首页请求，需要自行去除
 - 此测试没有 最终的报告形式，有效且完整的报告内容在后续章节说明

分析测试

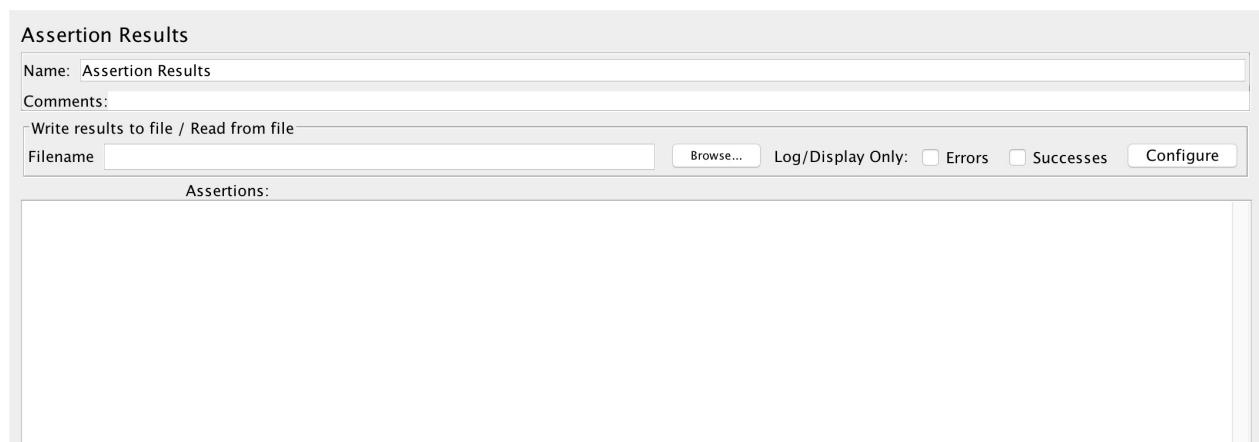
通过前三章的内容，应该可以 独立 进行一个项目的 简单 性能测试。But,要进行 完善 并 有效的性能测试，还需要继续深入学习后续的内容。后续的每个功能点，都会有完整例子。

- 断言结果详细分析
 - 断言结果 使用
 - 针对断言结果进行详细分析，快速定位出问题
- 结果详细分析
 - 概要结果 / 聚合报告 使用
 - 与 性能测试报告 的结合
- 服务器分析
 - 性能测试时，服务端关注点
- 扩展插件
 - 了解并应用扩展插件

断言结果详细分析

断言 经常被用于测试计划调试过程中，针对 断言结果 的分析可加倍测试调试效率。

Assertion Results

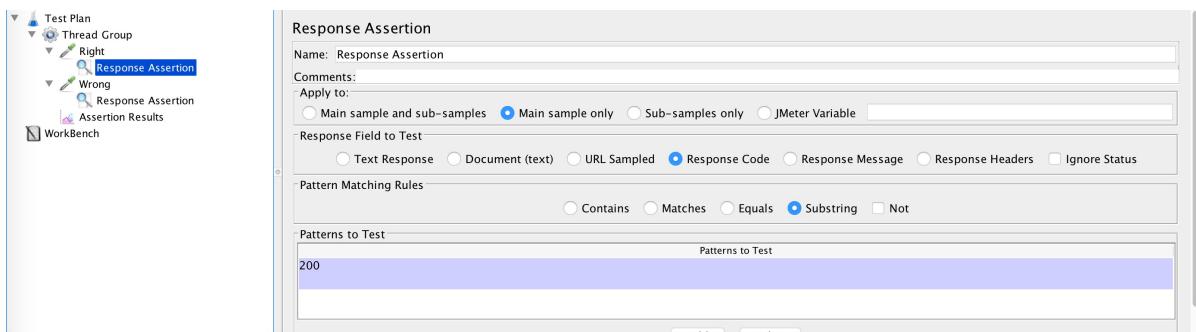


配制说明

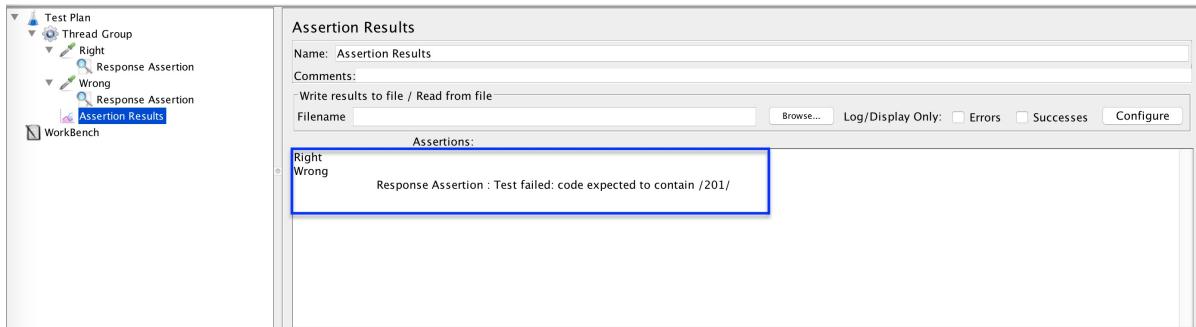
- Write results to file / Read from file : 保存结果或读取结果
 - Filename : 文件名， Browser 用于选择 目标文件
 - Log/Display Only : 日志/显示部分
 - Errors : 仅错误
 - Successes : 仅成功
 - Configure : 自定义部分，与其它结果可选项一致
- Assertions : 断言结果列表

实例

- 访问<http://www.jianshu.com/>会返回状态码 200
- 针对此 URL 请求，进行两次请求，且使用不同的状态码来 断言 : 200/201



- 执行测试，查看结果



- 在断言结果列表中: Right 请求正常 ; Wrong 请求下面的 Response Assertion 会断言失败，并显示 Response Assertion: Test failed: code expected to contain /201/`
- 完整示例参考: [AssertionResultsDetails.jmx](#)

应用

- 使用 Assertion 可以验证测试结果的 正确性
- 使用 Assertion Results 可以查看在整个测试过程中的所有 断言 结果，并回显具体的错误: 哪些在执行过程中出错，具体的错误内容是什么

结果详细分析

善于使用 JMeter 的结果分析报告功能，对于 性能测试 有很大的帮助。如：并发数据/响应数据/错误率

常用的报告有两种 `Summary Report` `Aggregate Report`，`Summary Report` 在第二章的 [结果分析](#) 中已讲过。

Aggregate Report

Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Error %	Throughput	KB/sec
TOTAL	0	0	0	0	0	0	0.922337203...	-92233720...	0.00%	.0/hour	.0

配制说明

- `Write results to file / Read from file` : 写结果到文件或读取文件内容
 - `File` : 需要加载的文件
 - `Log/Display Only`
 - `Errors` : 仅显示错误
 - `Successes` : 仅显示成功的部分
 - `Configure` : 需要配制保存的字段
- 结果展示列表
 - `Label` : 样例名
 - `# Samples` : 执行样例数
 - `Average` : 平均时间，单位：毫秒
 - `Median` : 50% 用户的响应时间，单位：毫秒
 - `90% Line` : 90% 用户的响应时间，单位：毫秒
 - `95% Line` : 95% 用户的响应时间，单位：毫秒
 - `99% Line` : 99% 用户的响应时间，单位：毫秒

- Min : 最小响应时间，单位: 毫秒
- Max : 最大响应时间，单位: 毫秒
- Error : 错误率
- Throughput : 吞吐量，默认情况下表示每秒完成的请求数(Request per Second)，单位: 秒
- KB/sec : 每秒 千字节
- Include group name in label : label 中显示线程组名
- Save Table Data : 保存 结果展示列表 中数据
- Save Table Header : 是否保存 列表头
- 示例结果

Aggregate Report												
Name: Aggregate Report												
Comments:												
Write results to file / Read from file												
Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Error %	Throughput	KB/sec	
HTTP Request	4	1354	1254	1764	1764	1764	951	1764	0.00%	2.3/sec	85.4	
TOTAL	4	1354	1254	1764	1764	1764	951	1764	0.00%	2.3/sec	85.4	

- 示例代码: [AggregateReport.jmx](#)

应用

- 测试结果数据，大部分 可直接体现在 测试报告 中

Summary Report VS Aggregate Report

属性对比

属性	SummaryReport	AggregateReport
Label	√	√
# Samples	√	√
Average	√	√
Min	√	√
Max	√	√
Std. DEV.	√	✗
Error %	√	√
Throughput	√	√
KB/sec	√	√
Avg. Bytes	√	✗
Median	✗	√
90% Line	✗	√
95% Line	✗	√
99% Line	✗	√

应用区别

- Summary Report 数据更侧重于在性能优化过程中使用，给团队内部看
- Aggregate Report 数据侧重于在产出最终的性能报告，给领导看

服务器分析

性能测试时，作为测试人员还需要关注服务器的数据，如: CPU/内存/网络。当然一份完整的性能测试报告，必然要包含 服务器 的数据结果，今天就来看下 服务器 常用的数据有哪些。

CPU

- 指性能测试场景运行的时间段内应用服务系统的 CPU资源占用率
- 判断系统 处理能力及应用运行 是否稳定的重要参数。
- CPU 使用率过高，则表示应用服务的 运算需求过高 。若持续较高时，需要针对 运算 部分进行优化

内存

- 指性能测试场景运行的时间段内应用服务系统的 内存使用
- 判断系统 处理能力及应用运行 是否稳定的重要参数。
- 内存 使用率过高，则表示应用服务器的 内存 中存储的数据量越大。若持续增涨，且回收不及时，则存在 内存泄漏 ，需要针对 对象回收 / 垃圾回收 进行优化

网络

- 指性能测试场景运行的时间段内服务器的网络 访问 返回 数据量
- 判断系统 数据优化 的重要参数。
- 网络 占用率过高，则表示传输过程中的数据量较大。若 数据量 过大，则需要针对传输数据进行 压缩 或针对数据进行 精简

磁盘IO

- 指性能测试场景运行的时间段内对主机硬盘的 读取 写入 数据量
- 判断系统 数据优化 的重要参数。
- 磁盘IO 过高，则表示针对磁盘的操作数据量较大。若 数据量 过大，则需要针对传输数据进行 压缩 或针对数据进行 精简

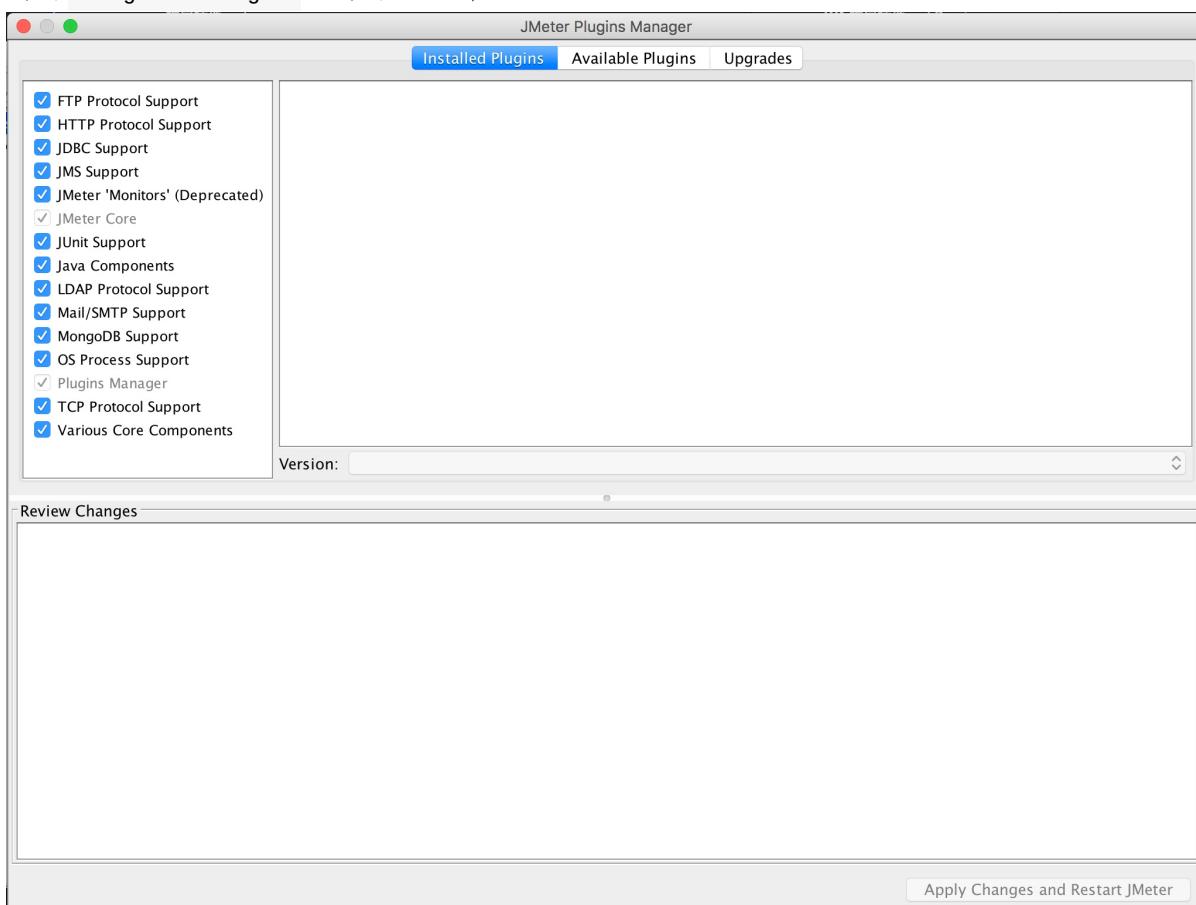
扩展插件

JMeter 虽然自带了一些丰富的功能，但在使用过程中，我们仍会有一些其它的需求。如：每个响应时间趋势图/服务器性能监视。本章内容主要围绕<https://jmeter-plugins.org/>官网的插件使用。

很多 JMeter 插件生成的结果，可直接用于性能测试报告中。如：Response Times Over Time

安装 Plugins-manager

- 下载 Plugins-manager，地址：[jmeter-plugins-manager-0.10.jar](https://jmeter-plugins.org/)。
- 将下载的文件 jmeter-plugins-manager-0.10.jar 复制到 JMeter 的文件夹 /lib/ext
- 重启 JMeter，在菜单选项 Options 最下方会多出一个功能 Plugins Manager
- 打开 Plugins Manager 会打开一个弹出框



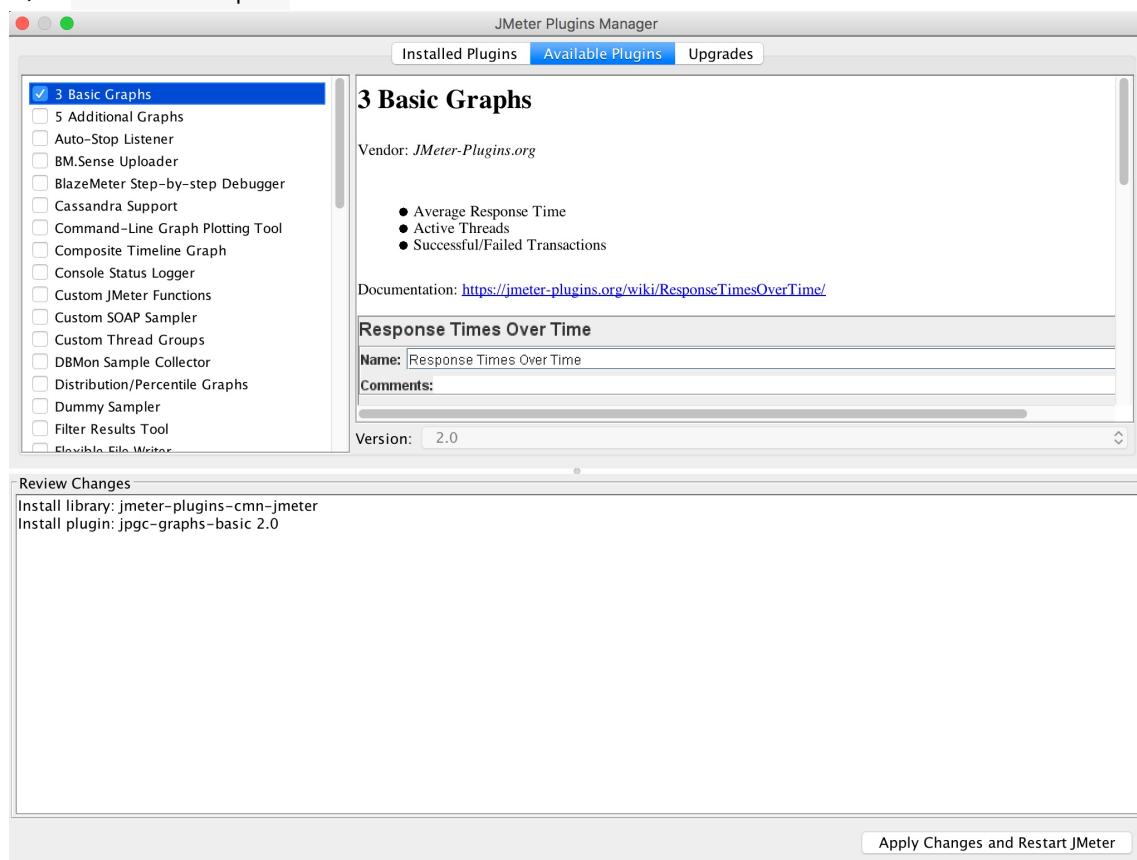
- Plugins-manager 安装成功

使用 Plugins-manager 安装 扩展插件

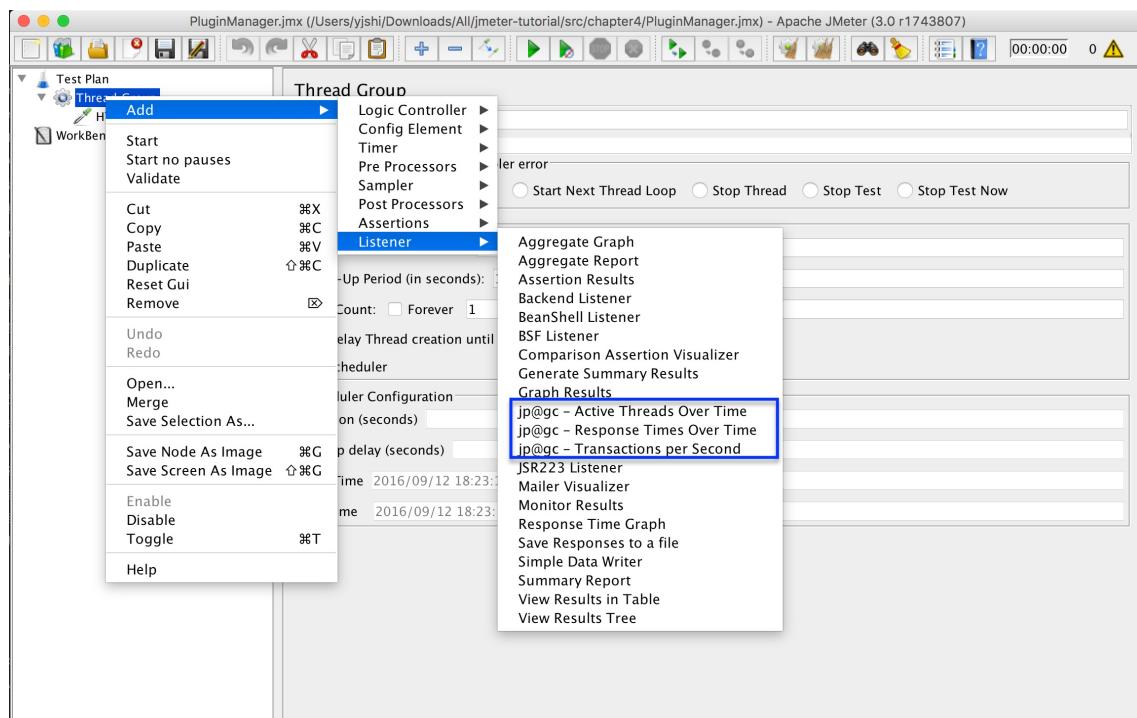
点击对应的插件名，可查看具体的插件 使用说明 。使用 3 Basic Graphs 插件进行举例说明。

- 安装插件 3 Basic Graphs

- 勾选 3 Basic Graphs

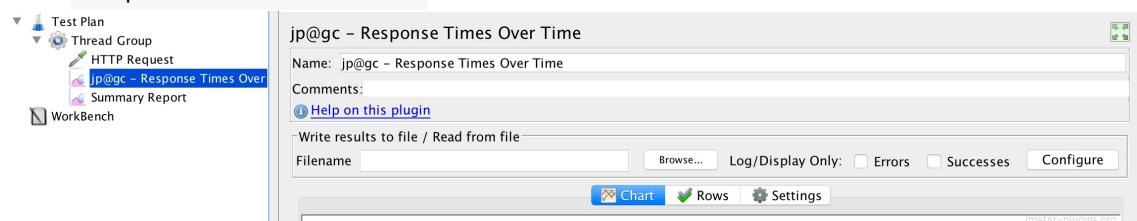


- 选择 Apply Changes and Restart JMeter
- 此时 JMeter 会通过插件管理 PluginsManager 去下载 3 Basic Graphs ，完成后自动重启 JMeter
- 在 Listener 中会多出 3 个可选项： Active Threads Over Time Response Times Over Time Transactions per Second

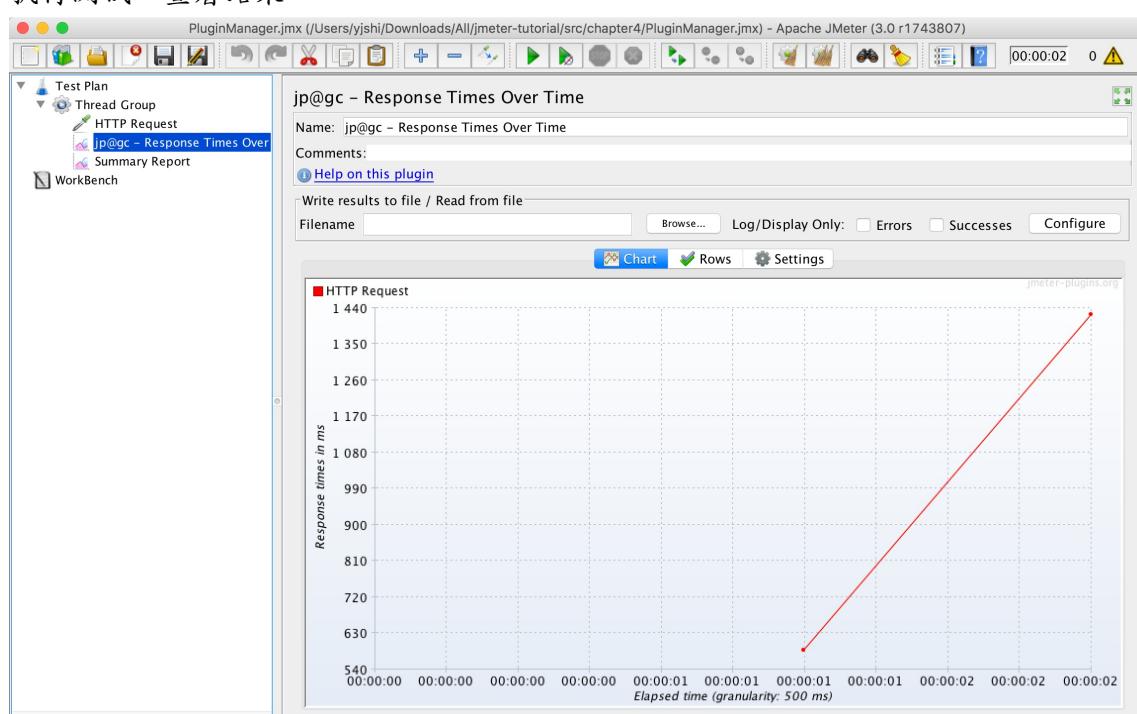


- 插件安装完成
- 使用插件

- 添加 Response Times Over Time



- 执行测试，查看结果



- 分析 Response Times Over Time

- 此插件主要用于展示测试过程中的 Response Time，并绘制成时序图

官方插件

JMeter-Plugins 官方给出了使用插件对应的 JMeter 版本及使用频率的插件列表。参考 <https://jmeter-plugins.org/stats/>

- JMeter 版本的使用情况



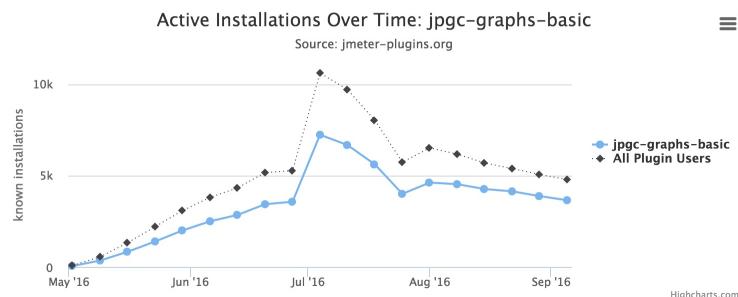
- 可以看出当前使用最多的为 JMeter 3.0 版本

- 各插件的使用情况

Plugins Popularity

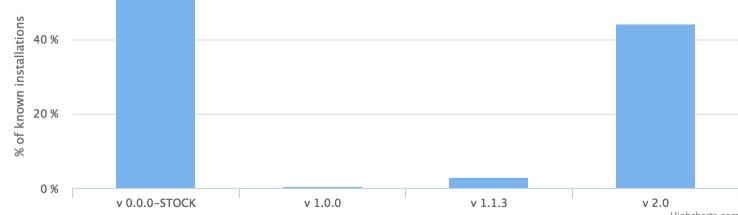
By the number of known users:

- Plugins Manager: 4779
- 3 Basic Graphs: 3647
- PerfMon (Servers Performance Monitoring): 3470
- Throughput Shaping Timer: 3408
- Dummy Sampler: 3284
- Flexible File Writer: 3261
- Custom JMeter Functions: 3224
- Inter-Thread Communication: 3219
- BM.Sense Uploader: 3196
- Composite Timeline Graph: 2911
- JSON Plugins: 2892
- 5 Additional Graphs: 2766
- Custom Thread Groups: 2765
- Console Status Logger: 2761
- Command-Line Graph Plotting Tool: 2761
- Merge Results: 2637
- jpcg - Standard Set: 2499
- Distribution/Percentile Graphs: 2482
- Variables from CSV File: 2454
- Graphs Generator Listener: 2402
- Parameterized Controller: 2330
- KPI vs KPI Graphs: 2316
- Synthesis Report: 2299
- HTTP Raw Request: 2258
- Test Plan Check Tool: 2248
- Auto-Stop Listener: 2184
- JMXMon Sample Collector: 2113
- Page Data Extractor: 2092
- DBMon Sample Collector: 2032
- Lock File Config: 1960
- OAuth Sampler: 1959
- UDP Protocol Support: 1948
- HTTP Simple Table Server: 1936



Installations by Version: jpcg-graphs-basic

Source: jmeter-plugins.org



- 插件分类

- 图形类
- 线程组
- 计时器

- 监听器
- 工具类
- 功能
- 逻辑控制
- 样例
- 配制项
- 前置操作
- 后置操作
- 断言

依据项目的需求找出 `合适的插件`，敢快应用到自己的项目中去吧。

进阶使用

- 命令行执行JMeter
 - 真实测试过程中，大多是需要使用 命令行 的方式来执行测试，快速了解使用命令行 执行测试
 - 部分常用参数功能
- 性能测试常用专业述语
 - 性能测试过程中的专业述语太多，列出测试过程中或别人口中的专业述语
- JMeter最佳实践
 - 官方最佳实践
 - WorkBench使用
 - 个人使用过程中的实践总结
- 性能测试最佳实践
 - 实际项目中最佳实践
- 测试报告
 - 结合JMeter快速生成 完成有效 的性能测试报告

命令行执行JMeter

使用 JMeter 在进行性能测试时，强烈推荐 使用命令行无**GUI**模式。

常用命令参数

参数名	用法
-n	指明使用无 GUI 模式运行 JMeter
-t	指定需要执行的测试计划源 文件名
-l	输出结果为 JTL 格式的文件名
-j	运行日志文件名
-r	使用 <code>remote_hosts</code> 中定义的 远程机器 来运行测试
-R	从命令行中指定 远程机器 来运行测试
-g	指定生成 csv 文件的结果报告存放位置
-e	在测试结束后生成报告
-o	在测试结束后生成测试报告的 文件夹 ，文件夹必须 存在或为空 。
-H	代理的 IP
-P	代理端口

- 应用

- `jmeter -t test.jmx` : 打开**JMeter**的**GUI**客户端，同时 打开 `test.jmx` 文件
- `jmeter -n -t test.jmx` : 使用非**GUI**模式，直接 进行测试。日志输出:

```

Creating summariser <summary>
Created the tree successfully using PerformMon.jmx
Starting the test @ Tue Sep 13 19:06:38 CST 2016 (1473764798835)
Waiting for possible Shutdown/StopTestNow/Heapdump message on port 4445
summary =      3 in 00:00:01 =    2.4/s Avg:   569 Min:   563 Max:   579 Err:
          0 (0.00%)
Tidying up ...    @ Tue Sep 13 19:06:40 CST 2016 (1473764800147)
... end of run

```

- `jmeter -n -t test.jmx -l test.jtl` : 使用非**GUI**模式，并将测试结果保存至 `test.jtl` 文件中
- `jmeter -n -r -t test.jmx` : 使用 `remote_hosts` 中配制的服务来执行 `test.jmx`

- `jmeter -n -R123.123.123.123,host1 -t test.jmx`：在 `123.123.123.123` 和 `host1` 中运行测试 `test.jmx`
 - `jmeter -n -t test.jmx -l log.jtl -H my.proxy.server -P 8000`：使用 `my.proxy.server` 的端口号 `8000` 来执行测试 `test.jmx`，并将结果保存至 `log.jtl` 文件

帮助

jmeter -h : 显示使用说明

Copyright (c) 1999-2016 The Apache Software Foundation

To list all command line options, open a command prompt and type:

jmeter.bat(Windows)/jmeter.sh(Linux) -?

To run Apache JMeter in GUI mode, open a command prompt and type:

imotor.bat (Windows) /imotor.sh (Linux) - Faz o property file

To run Apache JMeter in NON GUI mode:

Open a command prompt (or Unix shell) and type:

```
imometer.bat(Windows) /imometer.sh(Linux) -n -t test-file [-n property-file] [-l log-file]
```

To run Apache JMeter in NON_GUI mode and generate a report at end :
Open a command prompt (or Unix shell) and type:

```
jmeter.bat(Windows)/jmeter.sh(Linux) -n -t test-file [-p property-file] [-l log-file]  
-e -o [Path to output folder]
```

To generate a Report from existing CSV file:

Open a command prompt (or Unix shell) and type:

```
jmeter.bat(Windows)/jmeter.sh(Linux) -g [log-file] -o [path to output folder (empty or  
not existing)]
```

To tell Apache JMeter to use a proxy server:

Open a command prompt and type:

```
jmeter.bat(Windows)/jmeter.sh(Linux) -H [your.proxy.server] -P [your proxy server port  
]
```

To run Apache JMeter in server mode:

Open a command prompt and type:

```
jmeter-server.bat(Windows)/jmeter-server(Linux)
```

jmeter -? : 所有参数说明集合

```
--?  
    print command line options and exit  
-h, --help  
    print usage information and exit  
-v, --version  
    print the version information and exit  
-p, --propfile <argument>  
    the jmeter property file to use  
-q, --addprop <argument>  
    additional JMeter property file(s)  
-t, --testfile <argument>  
    the jmeter test(.jmx) file to run  
-l, --logfile <argument>  
    the file to log samples to  
-j, --jmeterlogfile <argument>  
    jmeter run log file (jmeter.log)  
-n, --nongui  
    run JMeter in nongui mode  
-s, --server  
    run the JMeter server  
-H, --proxyHost <argument>  
    Set a proxy server for JMeter to use  
-P, --proxyPort <argument>  
    Set proxy server port for JMeter to use  
-N, --nonProxyHosts <argument>  
    Set nonproxy host list (e.g. *.apache.org|localhost)  
-u, --username <argument>
```

```
Set username for proxy server that JMeter is to use
-a, --password <argument>
Set password for proxy server that JMeter is to use
-J, --jmeterproperty <argument>=<value>
Define additional JMeter properties
-G, --globalproperty <argument>=<value>
Define Global properties (sent to servers)
e.g. -Gport=123
or -Gglobal.properties
-D, --systemproperty <argument>=<value>
Define additional system properties
-S, --systemPropertyFile <argument>
additional system property file(s)
-L, --loglevel <argument>=<value>
[category]=level e.g. jorphan=INFO or jmeter.util=DEBUG
-r, --runremote
Start remote servers (as defined in remote_hosts)
-R, --remotestart <argument>
Start these remote servers (overrides remote_hosts)
-d, --homedir <argument>
the jmeter home directory to use
-X, --remoteexit
Exit the remote servers at end of test (non-GUI)
-g, --reportonly <argument>
generate report dashboard only, from a test results file
-e, --reportatendofloadtests
generate report dashboard after load test
-o, --reportoutputfolder <argument>
output folder for report dashboard
```

性能测试常用专业述语

列出性能测试中经常会用的一些 **述语**，供测试时参考

简称	全称	说明
RPS	Request Per Second	每秒请求数（吞吐量）
HPS	Hits Per Second	每秒点击次数，单位是次/秒
TPS	Transaction Per Second	每秒钟系统能够处理的交易或事务的数量
QPS	Query per Second	每秒处理查询次数，单位是次/秒
TPS	Transaction Per Second	每秒处理的事务数
PV	PageView	页面浏览量或点击量，用户每次刷新即被计算一次
UV	UniqueVisitor	访问的一台电脑客户端为一个访客
RT	Response Time	从客户端发一个请求开始计时，到客户端接收到从服务器端返回的响应结果结束所经历的时间，响应时间由请求发送时间、网络传输时间和服务器处理时间三部分组成。
VU	Virtual user	模拟真实业务逻辑步骤的虚拟用户
CPU	Central Processing Unit	指性能测试场景运行的这个时间段内，应用服务系统的CPU资源占用率
I/O	Input/Output	磁盘IO和网卡IO
JVM	Java Virtual Machine	Java虚拟机
GC	Garbage Collection	自动内存管理程序，是分配内存、保证被引用的对象始终在内存中、把不被应用的对象从内存中释放。 FGC 会引起JVM挂起
Network	-	数据传输速率，以Byte/s为单位
FR	Failure Ratio	错误率=(失败交易数/交易总数) * 100%

JMeter最佳实践

官方给出了一些使用 JMeter 的最佳实践，本文只对核心信息进行说明。JMeter 的另外一个功能 workBench 也是使用 JMeter 的一个很好功能，如果要使用 Recorder Controller，这个一定要会使用。

最后会给出一些个人在使用 JMeter 过程中的一些最佳实践。

官方最佳实践

- JMeter 官方给出了 15 条最佳实践，原文地址: [best-practices](#)，有兴趣的朋友可以自行去查看
- 针对官方的最佳实践，精要总结
 - 坚持使用最新版本，当前最新版本为 [3.0](#)
 - 使用一台机器并发时，不要将 并发数 设置过大
 - 利用 HTTP(S) Test Script Recorder 进行请求录制，提高脚本的维护效率
 - 将测试过程中所使用的 变量 进行统一管理，便于后期的维护
 - 真实测试过程中，最好使用 非GUI 模式进行测试
 - BeanShell 可以作为一个服务，用于快速调试
 - 尽量编写少的测试，将测试 通用化
 - 最好不要使用过时的方法

WorkBench 使用

WorkBench 是 JMeter 官方提供的一个用于 临时 存放一些数据。如: 复制请求及 部分元素 的应用。保存 测试计划 时，WorkBench 是不会被保存的，除非你手动选择保存。

上面说了 WorkBench 的功能，是不是感觉这东西没有啥用，But，JMeter 将一些特殊的元素应用于 WorkBench 中。如:

- **HTTP(S) Test Script Recorder:** 用于录制请求，如果要使用 Recorder Controller，一定 要有这个元素的配制
- **HTTP Mirror Server:** 微型镜像服务
- **Property Display:** 属性值的显示

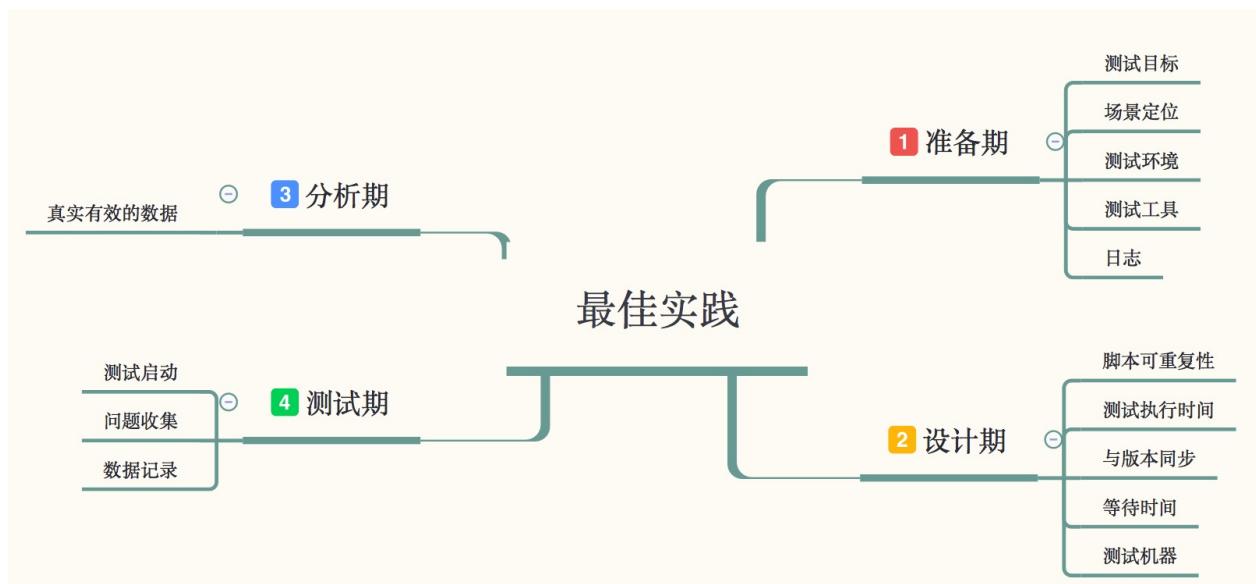
实践中的最佳实践

- 所有的变量最好统一来管理，建议使用 User Defined Variables
- JMeter 提供了一些内置的函数方法，在测试过程中会很有帮助。可在菜单栏中的 Function Helper 查看。常用的有
 - 获取本机IP: \${__machineIP()}
 - 随机数: \${__Random(,,)}
 - 随机生成字符串: \${__RandomString(,,)}
 - 计数器: \${__counter(,)}
 - 转码为Base64: \${__base64Decode(,)}
- 在当前打开的元素中按 `ctrl+H`，可直接查看对应元素的使用帮助文档
- 调试期，不要把数据量调的太大，很不便于分析结果
- 利用 JMeter 提供的 Controller 来完成 场景功能 的拆分

性能测试最佳实践

前面已经讲过 JMeter 使用最佳实践，以下再讲讲 通用性 的性能测试最佳实践。

依据性能测试 生命周期 不同，对每个 周期 内的最佳实践进行拆分。



准备期

- 测试目标
 - 明确性能测试的目标: 达成的数量级/基线数量级/场景覆盖
- 场景定位
 - 定位出需要测试的场景: 分析出 需要 进行性能测试的场景有哪些， 核心 的业务场景 有哪些
 - 不要假想测试场景: 不要 假想 一些可能会需要测试的场景，以免浪费测试时间
- 测试环境
 - 明确性能测试所测试的环境: 进行性能测试环境，要针对哪个环境进行。建议: 单独 搭建用于性能测试的环境，配制参考 生产 环境的配制。
 - 不要 直接使用 生产 环境进行性能测试，以免给生产环境造成影响。
- 测试工具
 - 针对测试工具的选取，一定要明确 所选取 的工具的 特性 ，不要盲目选择 听起来 很 NB的工具

- 测试工具，尽量选择一些 小巧/开源 ，够用即可
- 日志
 - 调整日志的 输出级别 ，减少日志输出对性能的影响
 - 日志中 最好 能输出对性能测试有帮助的日志

设计期

- 脚本可重复性
 - 性能测试脚本 最好 可 重复 执行，便于应对 不同的环境或版本更新
- 测试执行时间
 - 测试时间：设计测试执行时间时，一定要 适中 。强并发30分钟即可，持续操作2-3小时即可。
 - 测试周期：条件许可的情况下，最好在每个 小 迭代版本发布的时候都 至少 进行一次。至少 确保在上线前进行一次
- 与版本同步
 - 性能测试所涉及的内容，一定要与版本开发的内容同步更新。
- 等待时间
 - 性能测试过程中，在业务操作过程中 也 可加入 适当 的等待时间。 等待时间 在真实的 用户行为 中是 肯定 存在的
- 测试机器
 - 一定不要给客户端机器过大的压力，以免客户端机器性能不足时，影响整个测试结果。

测试期

- 测试启动
 - 测试开始时，尽量要减少 外部 因素对 测试环境 的影响。如：网络波动
- 问题收集
 - 及时收集问题：在测试过程中，若发现问题时，一定要及时保存 当时 的环境信息/日志文件/测试数据
- 数据记录
 - 测试过程中，确保针对每次的测试结果有准确的记录，便于后期的结果比对。

分析期

- 真实有效的数据
 - 分析性能测试结果时，一定基于 真实的 测试数据。确保数据的可靠性

常见性能问题

以下内容，不会涉及深入的 项目内部代码 分析，仅将性能测试过程中经常出现问题的点提出来，并给出对应的参考 定位方案 。

客户端性能

- 客户端测试机的性能不足，影响整个性能测试结果的真实性。在测试过程中，一定要先明确 客户端 测试机的物理配制及带宽情况，确保客户端机器有 足够 的能力来执行测试。
- 解决方案：若客户端性能不足，建议增加客户端机器的数量。

服务器内存泄漏

- 服务器对使用对象的回收不及或部分对象没有回收时，这是服务器内存泄漏的常见问题。但定位出具体哪部分有问题，还需要依据具体的项目去分析。
- 解决方案：输出 堆栈 信息，定位出具体的方法或对象问题。

数据库索引

- 数据库中创建的 索引过多 ，引起数据操作过慢，进而影响整个性能测试的时间。
- 解决方案：对性能测试场景中进行分析，找出哪些 数据操作 时长较高。优化数据库脚本

网络通信

- 服务器的网络数据未做 优化 时，带宽有 可能 会被消耗完全。带宽的消耗需要考虑 客户端 服务器 两方面，每一端的带宽均会影响测试数据的准确性。
- 解决方案：测试过程中，需要实时查看客户端/服务器的带宽数据，确保没有达到自身的数据上限。

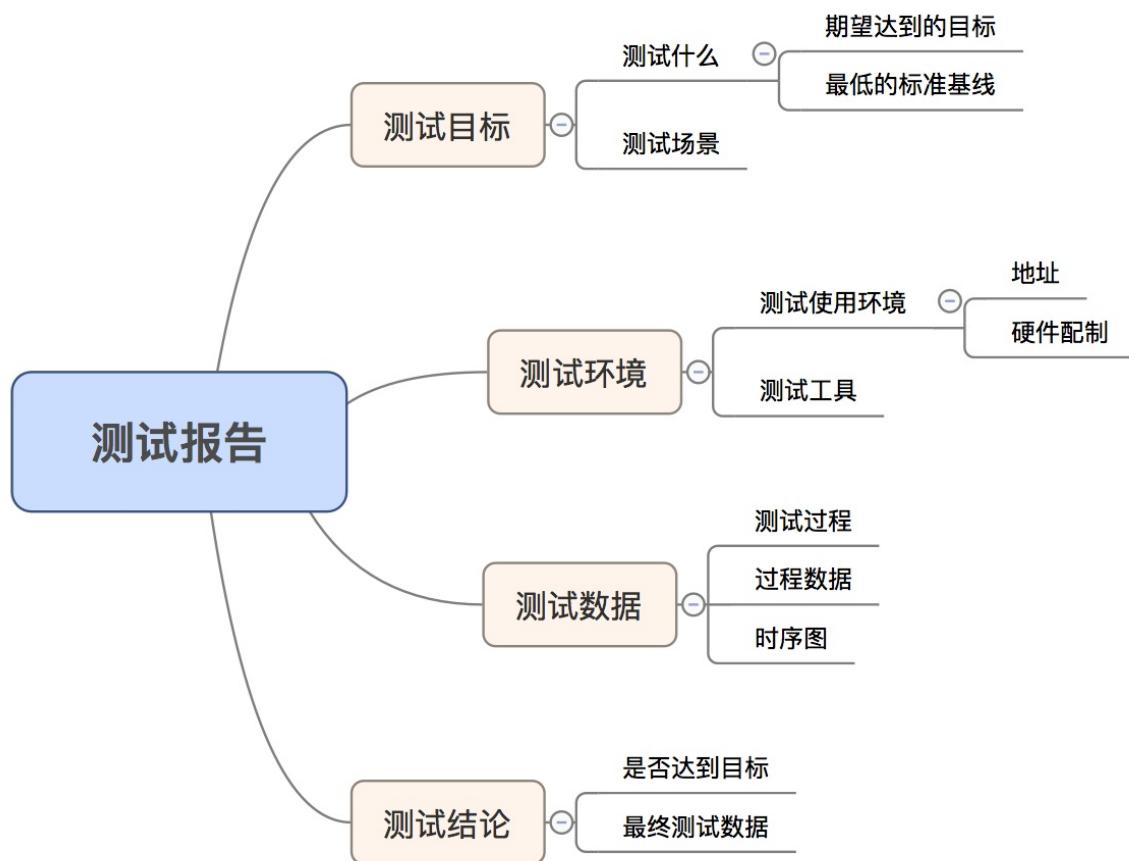
CPU较高

- CPU在测试过程中，经常会出现 居高不下或快速上升 的过程。大部分原因是由于服务端程序对CPU的计算量过大。
- 解决方案：尝试定位出是 哪些业务场景 引起CPU变量，再依据业务场景在 代码 中的处理方法定位出问题根源

测试报告

如果你已经看到这里，相信对性能测试已经有了一个完整 的认识了。那么接下来，你差的就是一个有效的 测试报告 。

测试报告内容



测试目标

- 明确测试目标是什么，要达到什么样的标准
 - 预期的测试数据 指标
 - 可接受的最低基线
- 测试场景
 - 明确哪些场景是进行测试的，有必要时，可说明为什么要选择 此场景 及对项目的影响

测试环境

- 测试使用的环境地址: 安装包/URL地址
- 测试使用的硬件配制
- 测试工具的选取

测试数据

- 测试过程, 进行的测试轮次/时间。也可直接给出 最终 的性能测试数据
 - 每个场景的测试过程数据
 - 数据是否需要更加完善的 时序图 来展示

测试结论

- 是否达到了 预期 的指标
- 最终的 数据概述

结束语

哇，你竟然看到最后了，看来你真的很想把性能测试做好。那就再送你点 建议

- 测试过程中，如果出现脚本 调试不通或者难以调试 时，记得 及时 请教开发人员
- 针对有些可重复利用的数据，尽量要 好好保存 。减少不必要的 重复劳动
- 不要过于依赖 工具 ，有时电脑自身的功能就可以满足你。如：
 - Windows 的 资源监视器 /Unix 中的 TOP 就提供了常用的性能数据
 - 工具统计的数据会 额外 占用你的机器资源

最后，希望你性能测试一切顺利！