

# JMeter 性能与接口测试

# 课程目录

- JMeter 概要介绍
- JMeter 环境搭建
- JMeter 目录结构及常用元件
- JMeter 录制方法介绍
- JMeter 参数化
- JMeter事务和集合点
- JMeter 检查点
- JMeter 监听器
- 非GUI模式下运行JMeter
- JMeter案例与实践



# 性能测试定义

- 性能测试 (Performance Testing):

在一定的负载情况下，系统的响应时间等特性是否满足特定的性能需求

- 区分以下用户数概念:

- 在线

- 并发

- 预计系统实际用户数

# 性能测试类型-1（按测试目的不同）

- 负载测试（Load Testing）：

负载测试关注的是不同负载水平的系统的性能指标。为了得到压力数确定下的性能指数。例如我们可以找到系统的最大用户数和最佳用户数。

- 压力/强度测试（Stress Testing）：

压力测试则是关注在超高负载（超过系统最大用户数）的情况下，系统是否还能稳定处理，如果不能稳定处理，那么系统还能坚持多久。其重点是关注系统是如何失效的，以便于制定系统正式上线后的风险控制措施。

# 性能测试类型-2

- 配置测试（Configuration Testing）：

在不同的软件、硬件以及网络环境配置下，通过运行一种或多种业务在一定的虚拟用户数量情况下，**获得不同配置的性能指标**，用于选择最佳的设备及参数配置。

- 容量测试（Volume Testing）：

目的是通过测试预先分析出软件中**某指标的极限值**（如最大并发用户数、数据库记录数等），保证系统在其极限状态下没有出现问题并能正常运行。

# 性能测试类型-3

- **基准测试（Benchmark Testing）：**

在一定的软件、硬件及网络环境下，模拟一定数量虚拟用户运行一种或多种业务，**将测试结果作为基线数据**，在**系统调优**或者系统评测过程中，通过运行相同的业务场景并比较测试结果，**确定调优是否达到效果**或者为系统的选择提供决策数据。

- **并发测试（Concurrency Testing）：**

通过模拟**多个用户并发访问**同一个应用、同一个存储过程或数据记录以及其他并发操作，测试是否存在死锁、数据错误等故障。

# 性能指标的介绍

- 工作负荷 = 虚拟用户数

对服务器产生多大压力，可以由多少用户同时对服务器发送请求来衡量。

- response time 响应时间

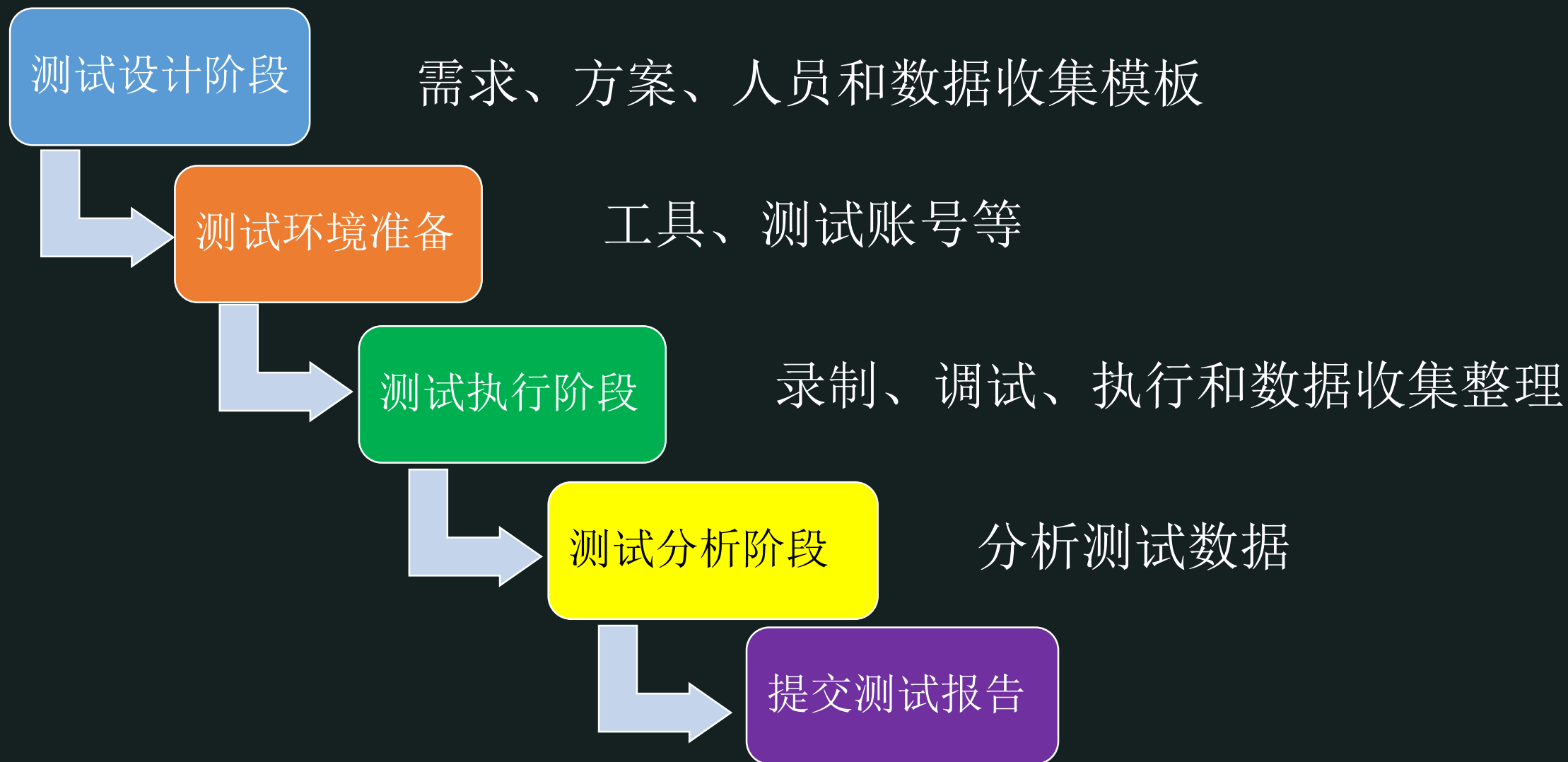
从客户端将数据包发出，到接收到服务器端发来的响应。

- throughput ~Ti & To 吞吐量 吞吐量越大表示系统性能越强。

- Hits/Request 网页点击数/请求

- Hits Per Second 每秒种点击次数

# 性能测试实施





# JMeter 概要介绍

- JMeter 简介
- JMeter 特点
- JMeter工作原理
- JMeter VS LoadRunner



# JMeter 简介

- Apache JMeter是Apache组织开发的基于Java的**压力**测试工具，同时又是一款**接口**测试工具。
- JMeter可以用于对服务器、网络或者对象模拟巨大的**负载**，来自不同压力类别下测试他们的**强度**和分析整体**性能**；
- JMeter能够对于应用程序做**功能/回归**测试，通过创建带有断言的脚本来验证你的程序返回了你期望的结果；

# JMeter 特点

- 能够对HTTP和FTP服务器进行压力和性能测试，也可以对于任何数据库进行同样的测试。
- 免费开源，纯Java开发的性能测试工具。
- 应用：静态文件、java小服务程序、CGI脚本、Java对象、数据库、FTP服务器、邮件服务器等。
- 各种负载统计表和计时器可供选择；
- 数据分析和可视化插件提供了很好的可扩展性以及个性化；
- 具有提供动态输入到测试的功能；

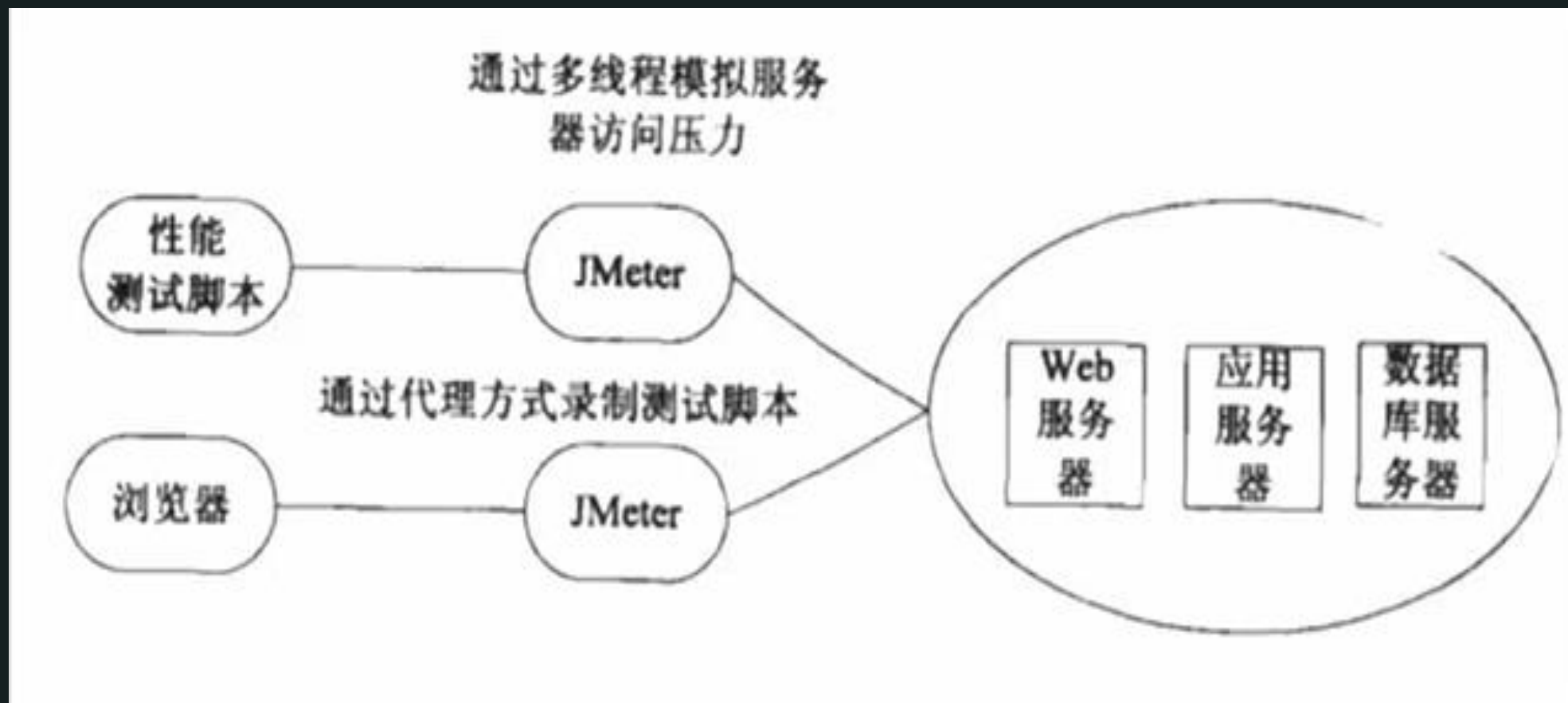
# JMeter VS LoadRunner

对比项	JMeter	LoadRunner
安装	简单，下载解压即可	复杂，LoadRunner 安装包大于 1GB，在一台主频 3.0、内存 1GB 的 PC 上安装，安装时间大于 1 小时
录制/回放模式	支持	支持
测试协议	偏少，用户可自行扩展	较多，用户不可自行扩展
分布式大规模压力测试	支持	支持
IP 欺骗功能	不支持	支持
图形报表	支持（较弱）	支持（很强）
测试逻辑控制	支持	支持
监控服务器资源（CPU、内存等）	支持	支持
功能测试	支持	不支持

# JMeter VS LoadRunner

- 版权问题
- Jmeter作为开源的性能测试工具，能够实现Loadrunner95%以上的功能。
- Jmeter支持二次开发，能够针对企业产品做调整，能够更好的满足企业性能测试需求。
- 缺点：
  - 用户友好性及集成监控不如Loadrunner
  - 本身的性能不如Loadrunner

# JMeter 工作原理



# 课程目录

- JMeter 概要介绍
- JMeter 环境搭建
- JMeter 目录结构及常用元件
- JMeter 录制方法介绍
- JMeter 参数化
- JMeter事务和集合点
- JMeter 检查点
- JMeter 监听器
- 非GUI模式下运行JMeter
- JMeter案例与实践



# JMeter 环境搭建

## 1. 前置条件

安装JDK，建议是JDK8以上版本，  
下载地址

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

## 2. JMeter下载

JMeter下载地址：

[http://jmeter.apache.org/download\\_jmeter.cgi](http://jmeter.apache.org/download_jmeter.cgi)

## 3. 启动JMeter

直接解压，无需安装

找到bin目录里的jmeter.bat双击即可启动

(如： C:\tools\apache-jmeter-5.0\bin\jmeter.bat)

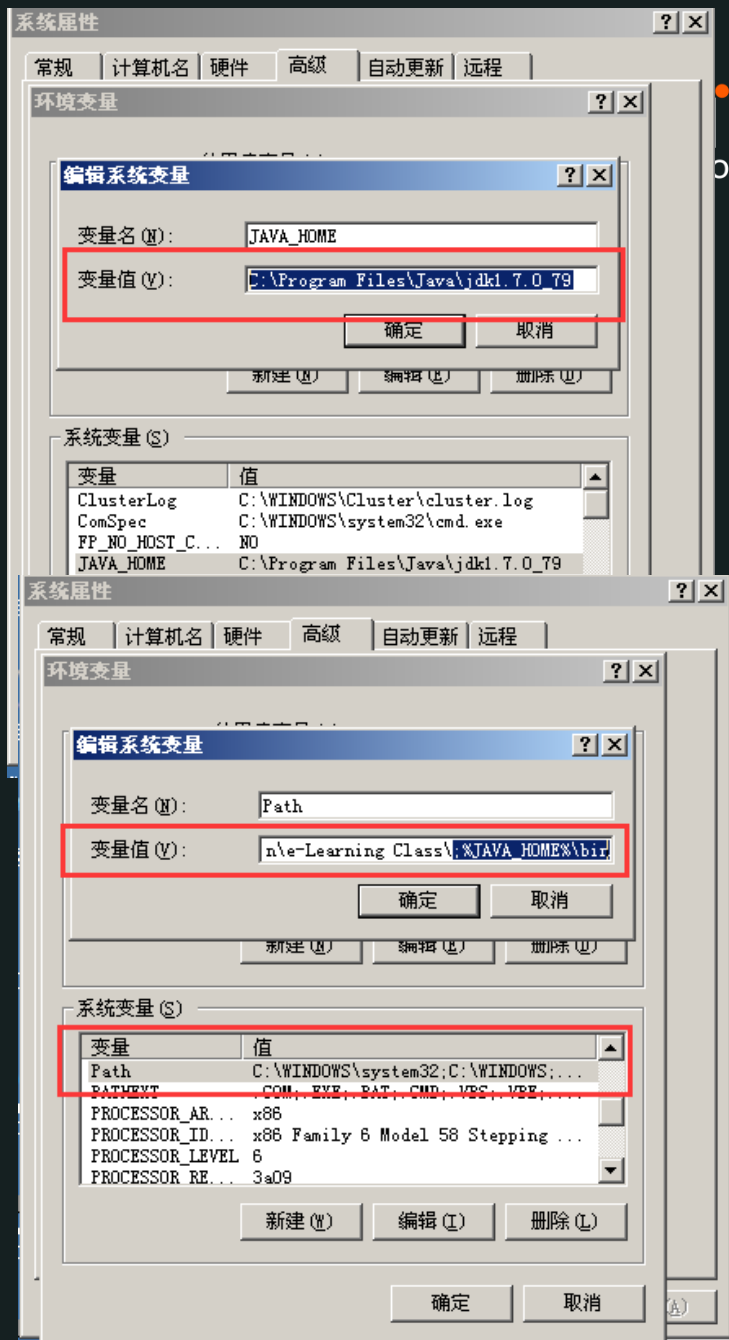


# 环境变量配置

- 1、右键我的电脑-属性-高级-环境变量
- 2、在系统变量内新增一个变量, JAVA\_HOME, 变量值为:

C:\Program Files\Java\jdk1.8.0\_191

- 3、需要修改系统变量path, 在变量最后添加 ;%JAVA\_HOME%\bin



# 入门演示

- 入门演示：论坛的发帖
- 目的：
  - 完成一个测试环境的搭建
  - 对软件界面的初步认识
  - 了解一个录制和回放的过程
  - 观察并“用心”去体会



# 课程目录

- JMeter 概要介绍
- JMeter 环境搭建
- **JMeter 目录结构及常用元件**
- JMeter 录制方法介绍
- JMeter 参数化
- JMeter事务和集合点
- JMeter 检查点
- JMeter 监听器
- 非GUI模式下运行JMeter
- JMeter案例与实践



# JMeter 目录结构

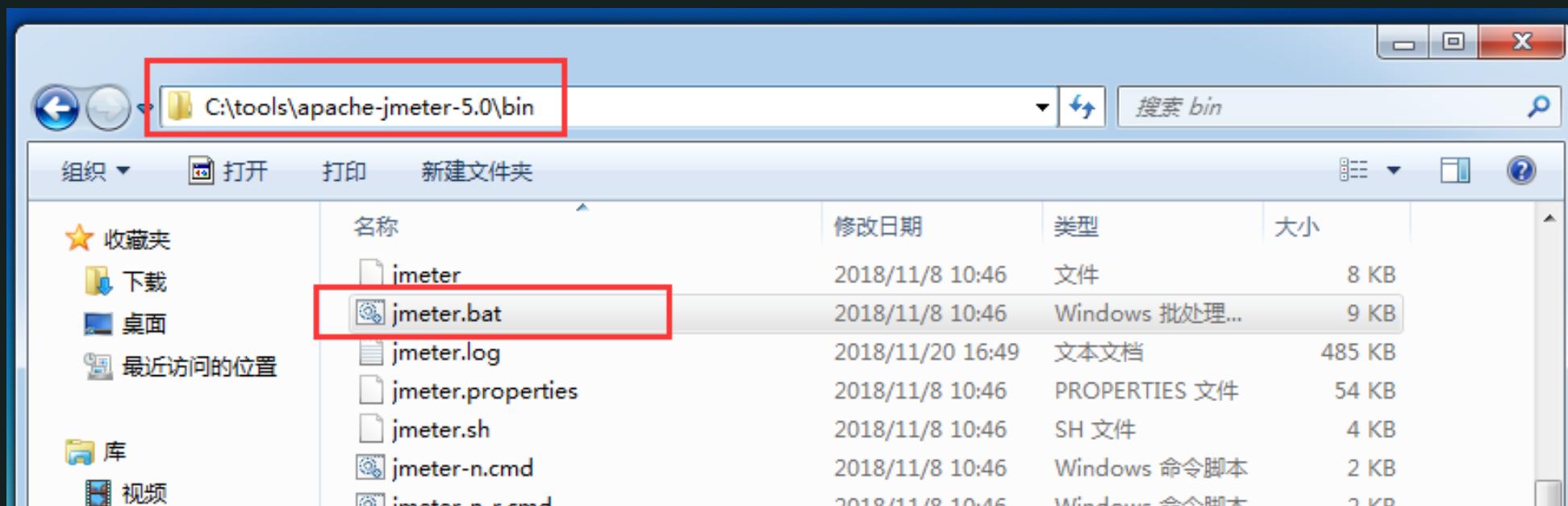
- bin目录：可执行文件，**jmeter.bat**启动
- docs目录：API文档目录，二次开发用
- extras目录：扩展插件目录，  
目录下的文件提供了对ant的支持。
- lib目录：所用到的插件目录，里面全是jar包，  
用户扩展所依赖的包直接放到lib下即可
- printable\_docs/usermanual子目录：  
jmeter用户手册，其中component\_reference.html  
是最常用的核心元件帮助手册；
- lib/ext子目录：jmeter核心jar包；



# Jmeter的启动文件

## Jmeter.bat

- Jmeter.bat是Jmeter的主运行程序，我们通过这个程序启动Jmeter。

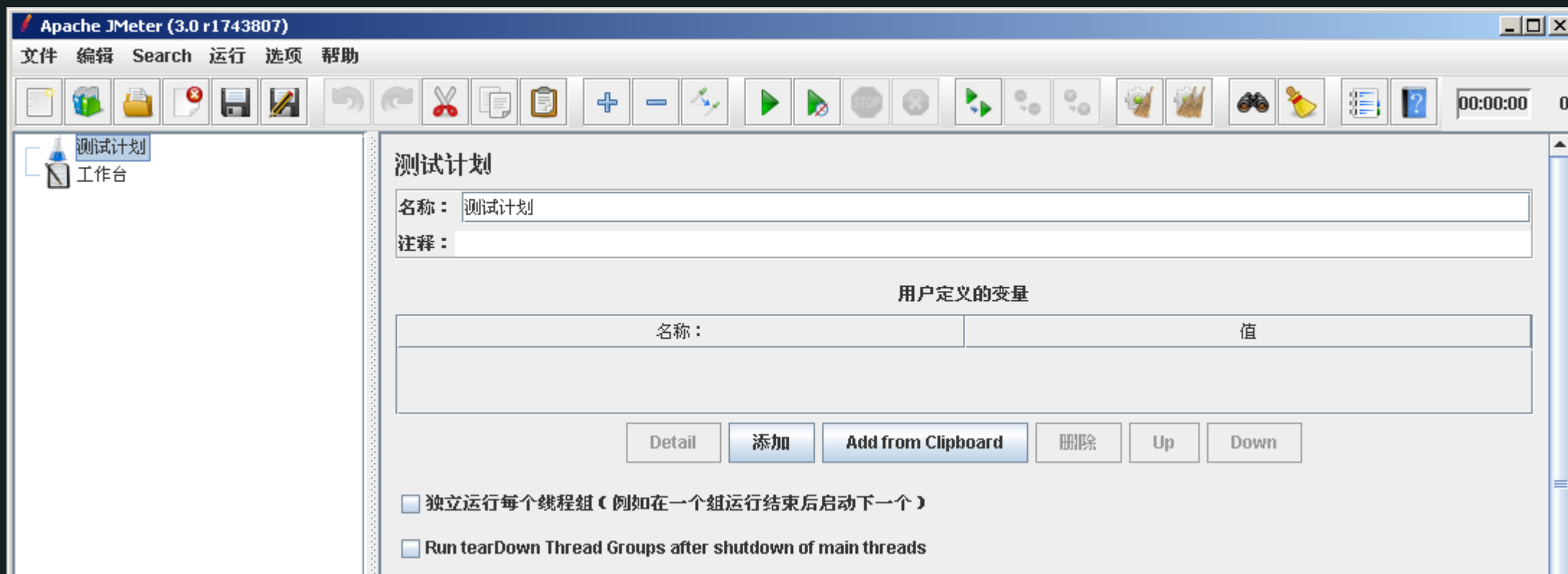


# JMeter 常用元件

- 测试计划
- 线程（用户）
- 测试片段
- 控制器（取样器，逻辑控制器）
- 配置元件
- 定时器
- 前置处理器
- 后置处理器
- 断言
- 监听器

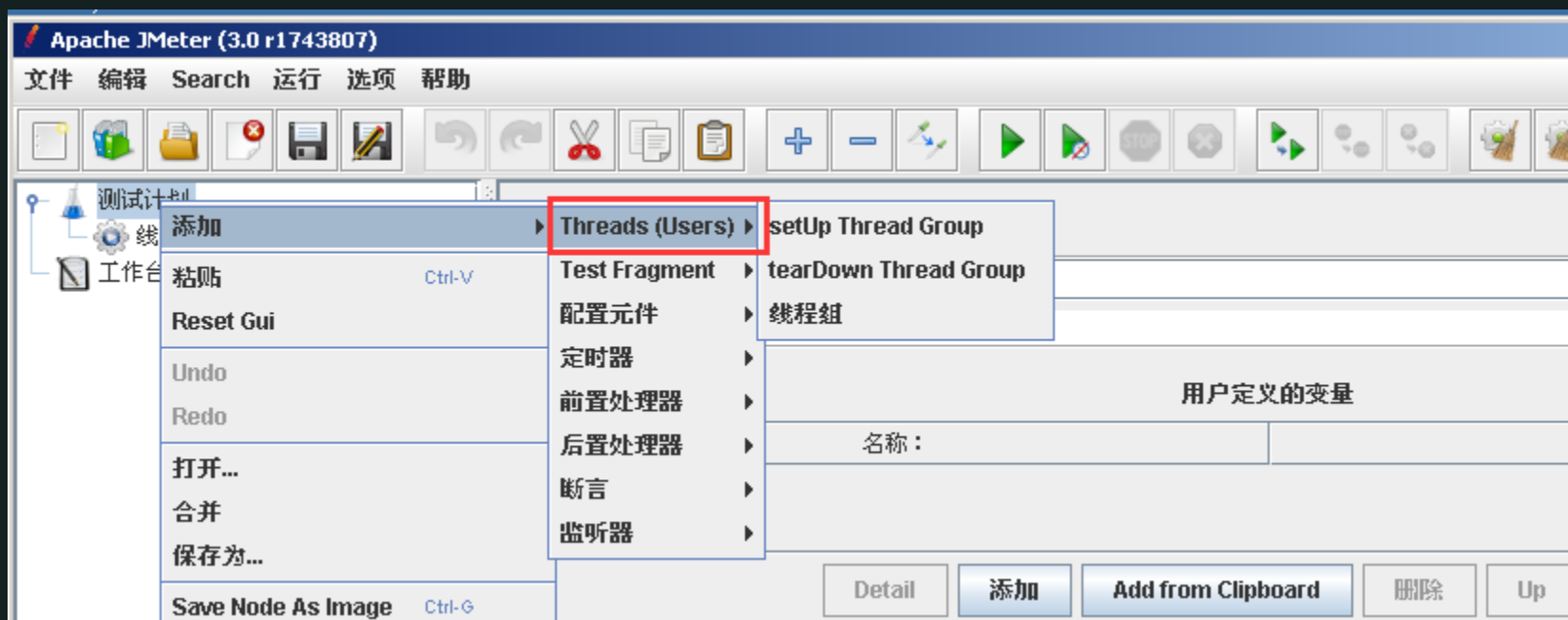
# 测试计划 (Test Plan)

- 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。
- 本次测试的所有内容都是基于一个计划的。



# Threads (Users)

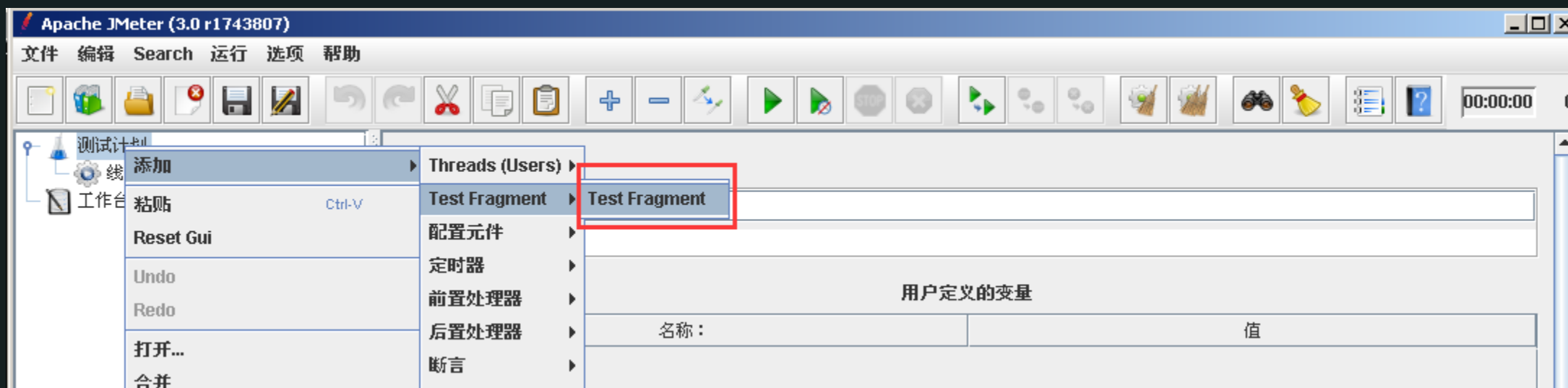
- 一个线程组，可以看做一个虚拟用户组，线程组中的每个线程都可以理解为一个虚拟用户。
- 线程组中包含的线程数量在测试执行过程中是不会发生改变的。





# 测试片段 (Test Fragment)

- 同级于线程组。
- 可包含完整的业务请求，但不能定义用户数。
- 不能单独执行，可以被模块控制器调用执行。

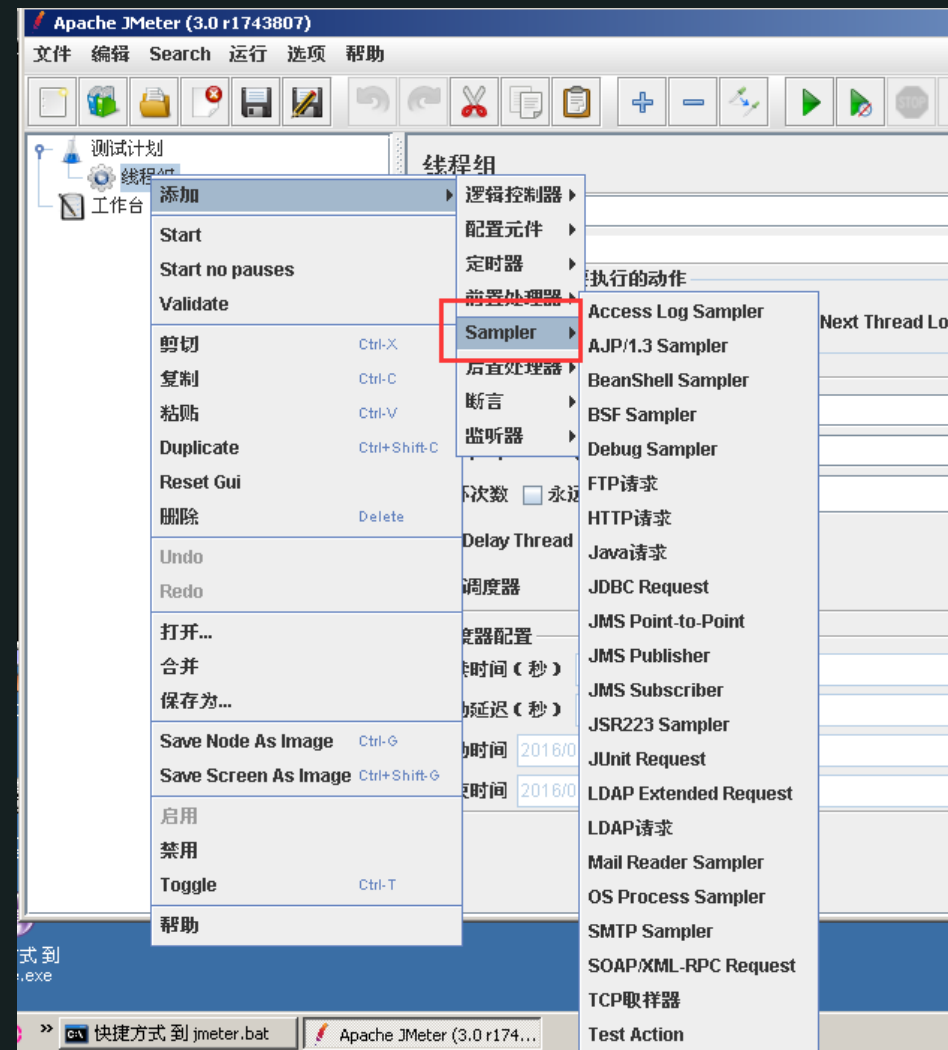


# 控制器

用途：驱动处理一个测试。

两种控制器：

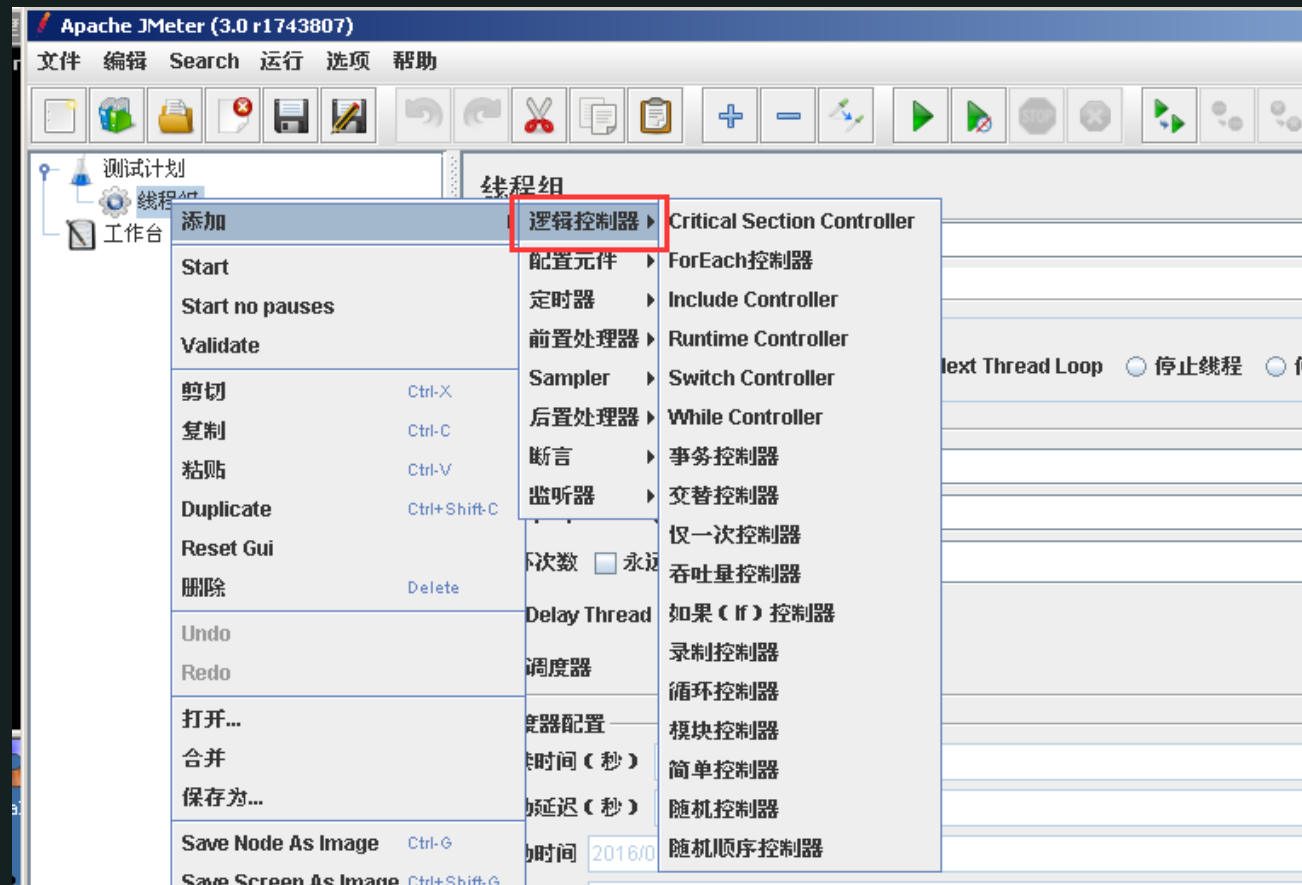
- 取样器（Sampler）是性能测试中向服务器发送请求，记录响应信息，记录响应时间的最小单元；



# 控制器

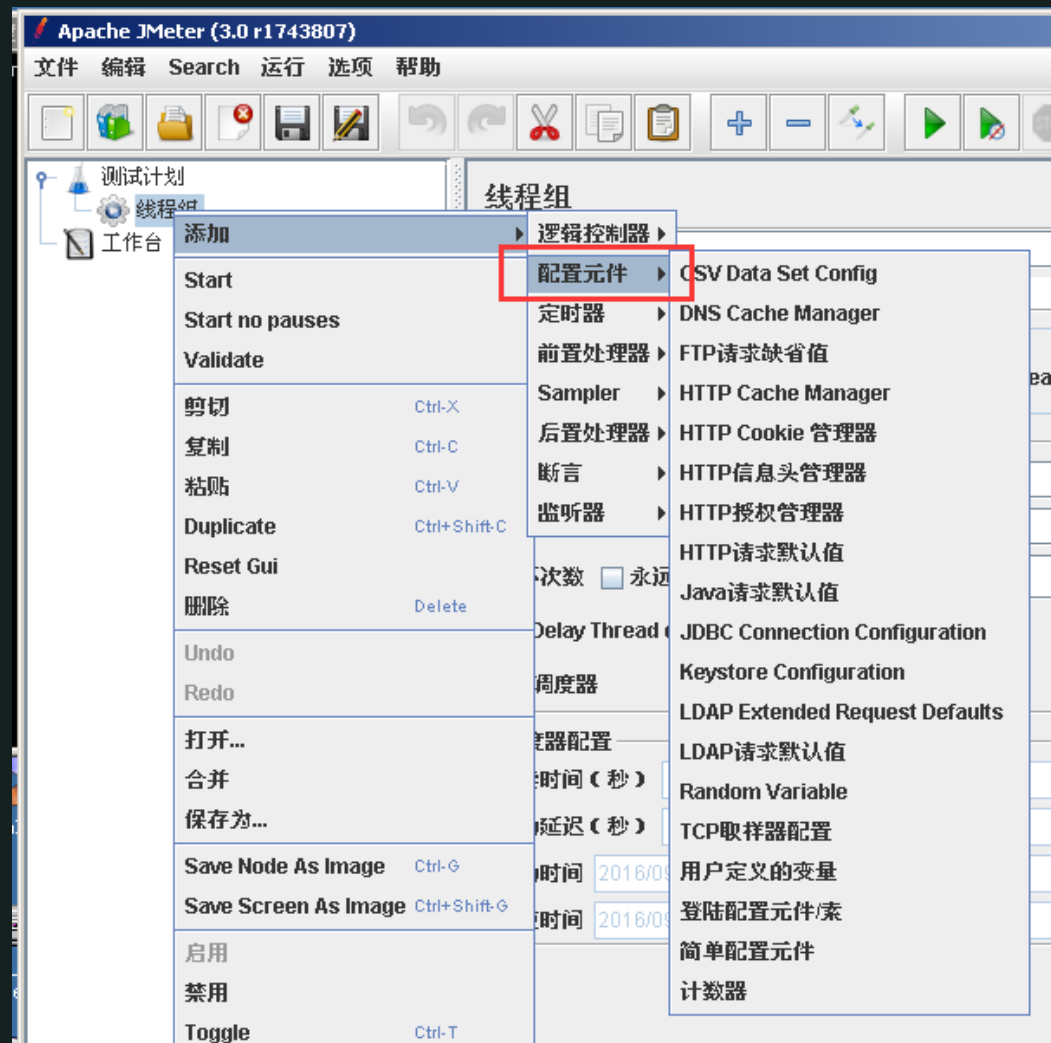
- 逻辑控制器

- 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。
  - ✓ 用来组织sampler 节点的，如事务控制器、吞吐量控制器。



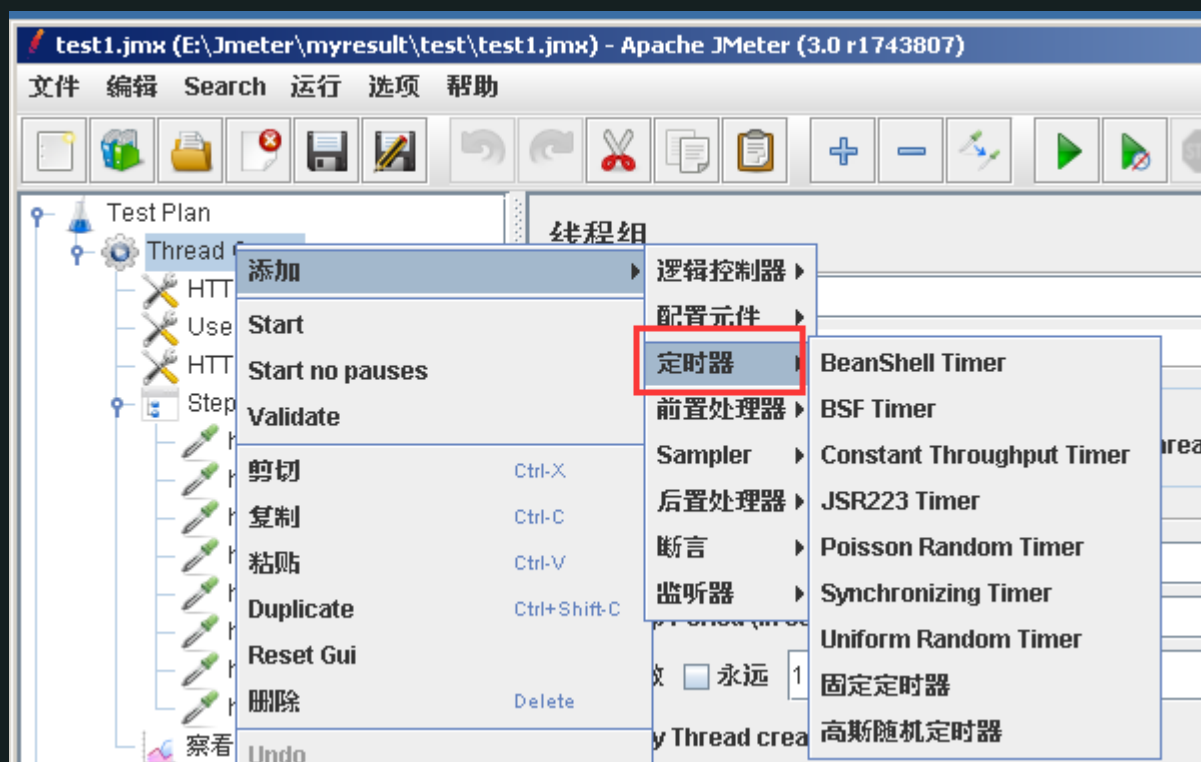
# 配置元件（Config Element）

- 作用：提供对静态数据配置的支持。
- 例如：
  - CSV Data Set config 可以将本地数据文件形成数据池（用作参数化）
  - HTTP Cookie Manager 可以用于对 HTTP Request Sampler 的 cookie 进行管理



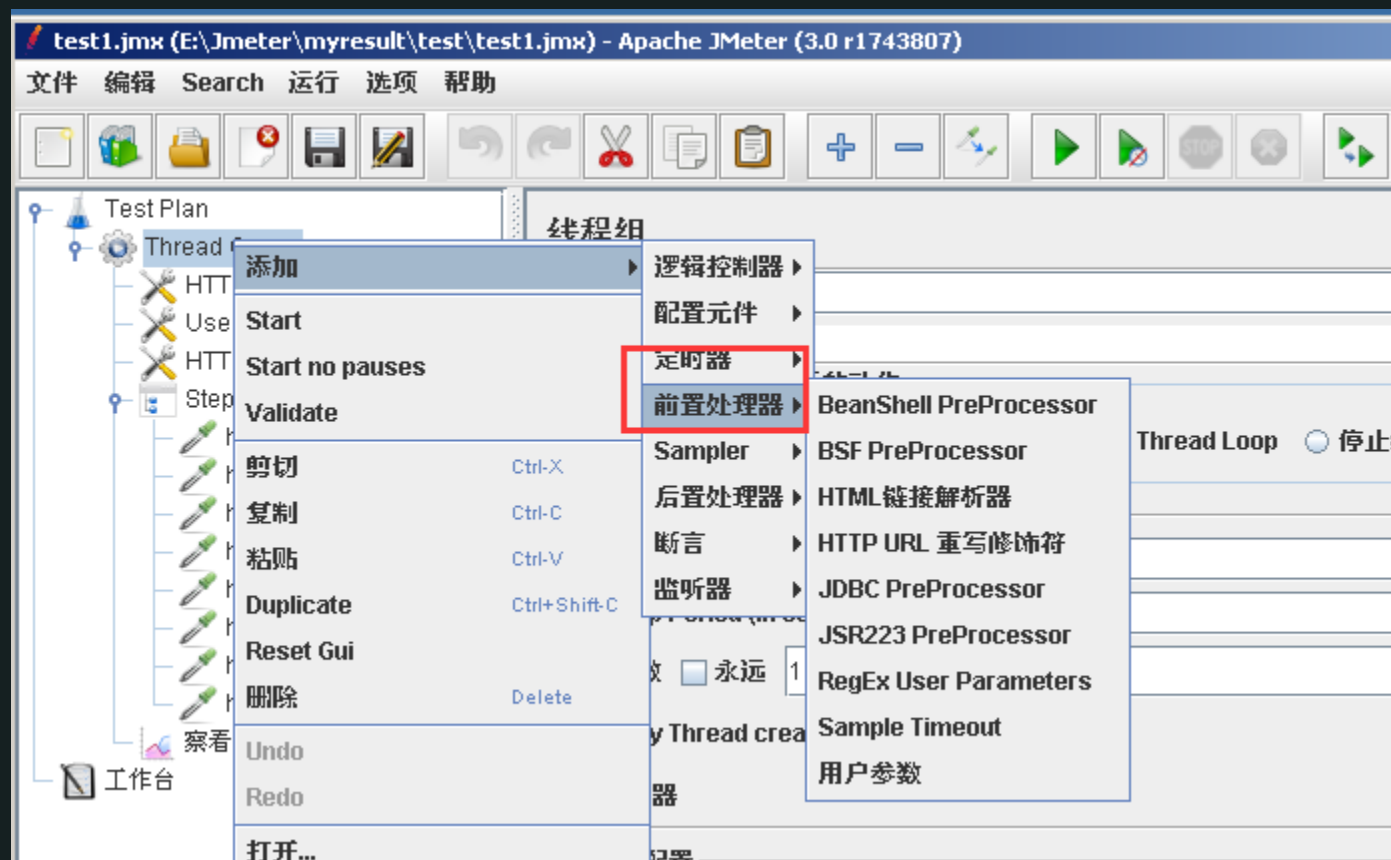
# 定时器 (Timer)

- 作用：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的手段。
- 例如：思考时间、集合点



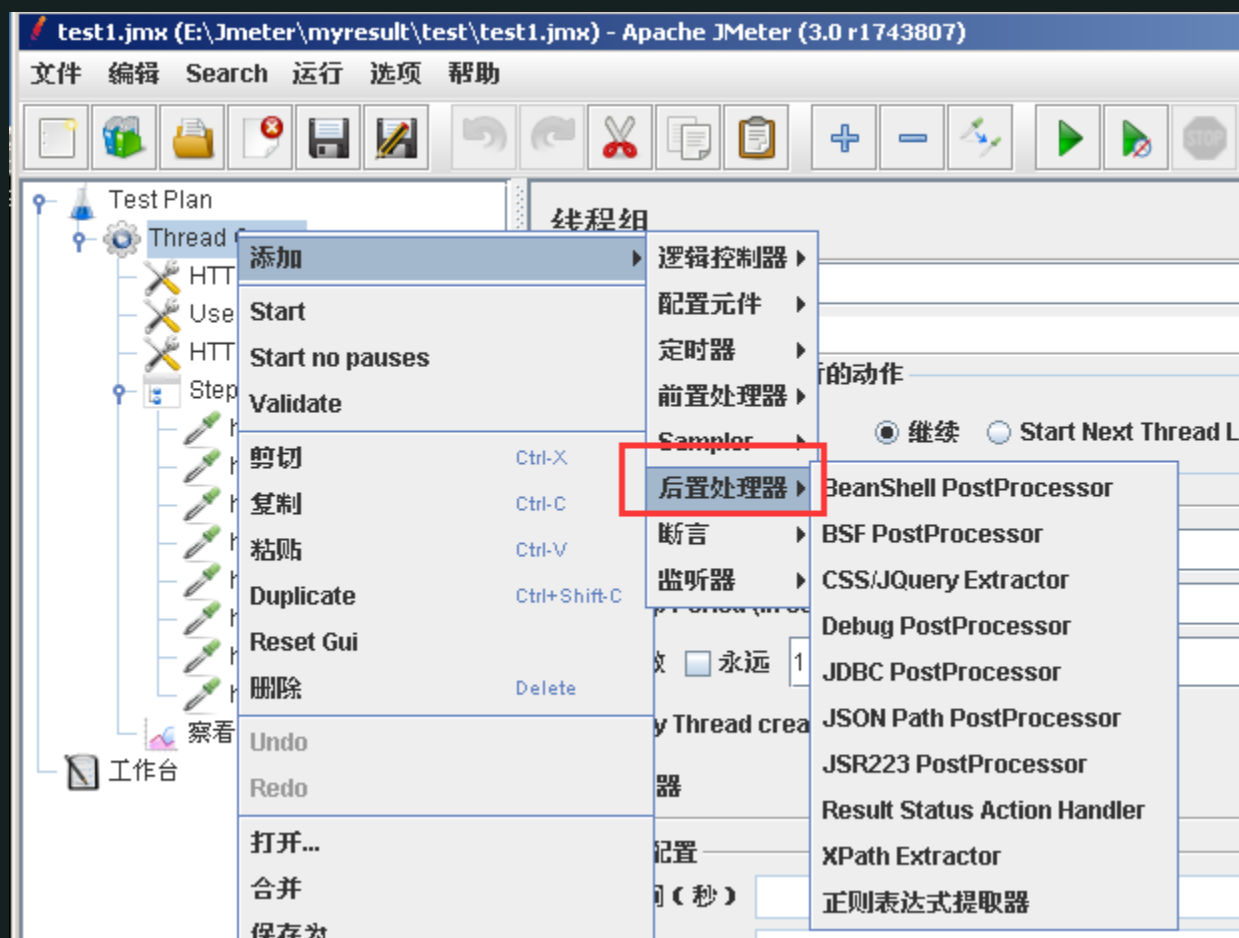
# 前置处理器 (Per Processors)

- 作用：在实际的请求发出之前对即将发出的请求进行特殊处理。



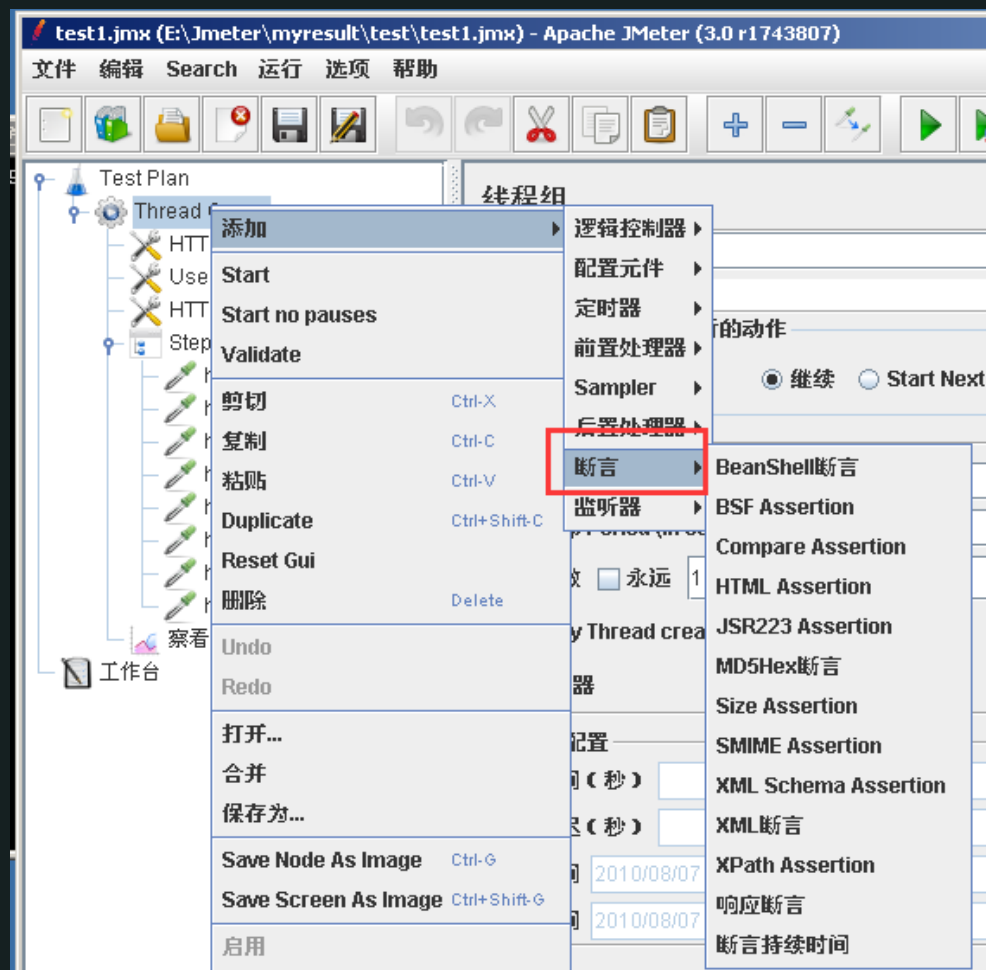
# 后置处理器 (Post Processors)

- 作用: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



# 断言 (Assertions)

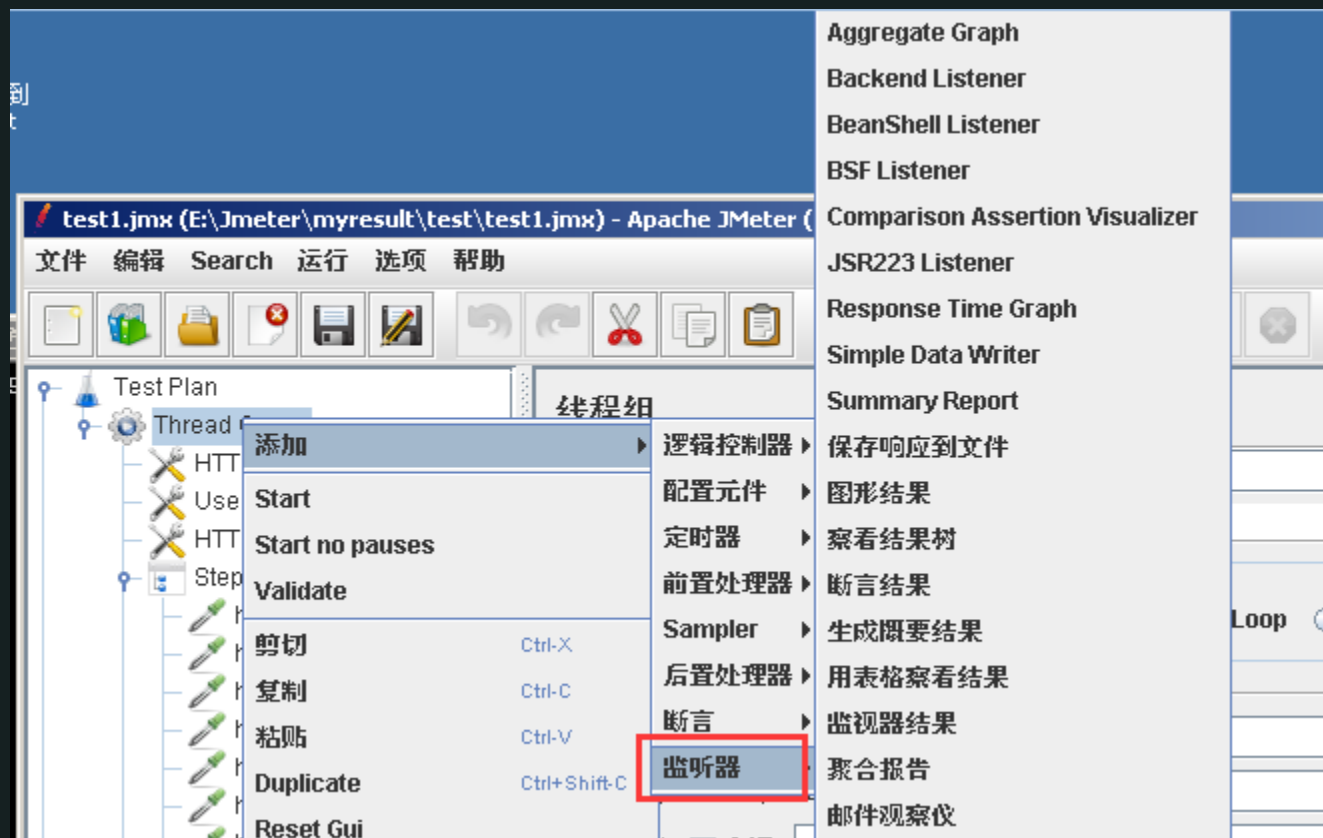
- 作用: 检查测试中得到的相应数据等是否符合预期, 断言一般用来设置检查点, 用以保证性能测试过程中的数据交互是否与预期一致。





# 监听器 (Listener)

- 作用：对测试结果数据进行处理和可视化展示。
- 例如：图形结果、察看结果树、聚合报告、用表格察看结果等



# JMeter 元件作用域与执行顺序

- HTTP1,2,3无作用域的概念
- 循环控制器：作用域HTTP2,3，以及图形结果
- 图形结果：作用域HTTP2,3
- 聚合报告：作用域HTTP1,2,3



# JMeter 元件作用域与执行顺序

- 循环控制器：作用域HTTP2,3，图形结果，随机控制器
- 响应断言：作用域JDBC
- 聚合报告：作用域 所有



# JMeter 元件作用域与执行顺序

- 固定定时器：
  - 对和它同级的和下级都有效
  - 同级中不分先后顺序
  - 一个取样器同时受2个固定定时器控制，控制的时间为两个定时器之和
- 固定定时器演示

# 课程目录

- JMeter 概要介绍
- JMeter 环境搭建
- JMeter 目录结构及常用元件
- **JMeter 录制方法介绍**
- JMeter 参数化
- JMeter事务和集合点
- JMeter 检查点
- JMeter 监听器
- 非GUI模式下运行JMeter
- JMeter案例与实践



# JMeter 录制方法

- 第三方BadBoy录制



- JMeter代理录制



- 手动编写

# BadBoy录制

- 过程：

- New test

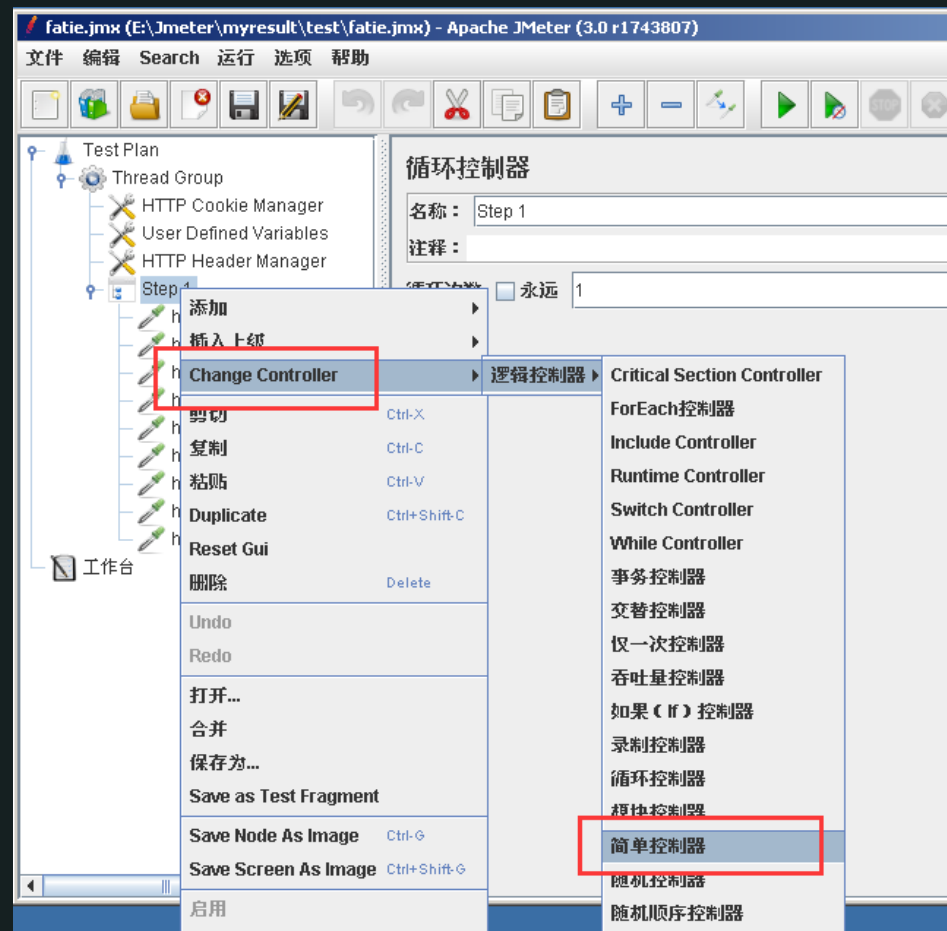
- New step

- 录制、暂停按钮

- File-export to Jmeter

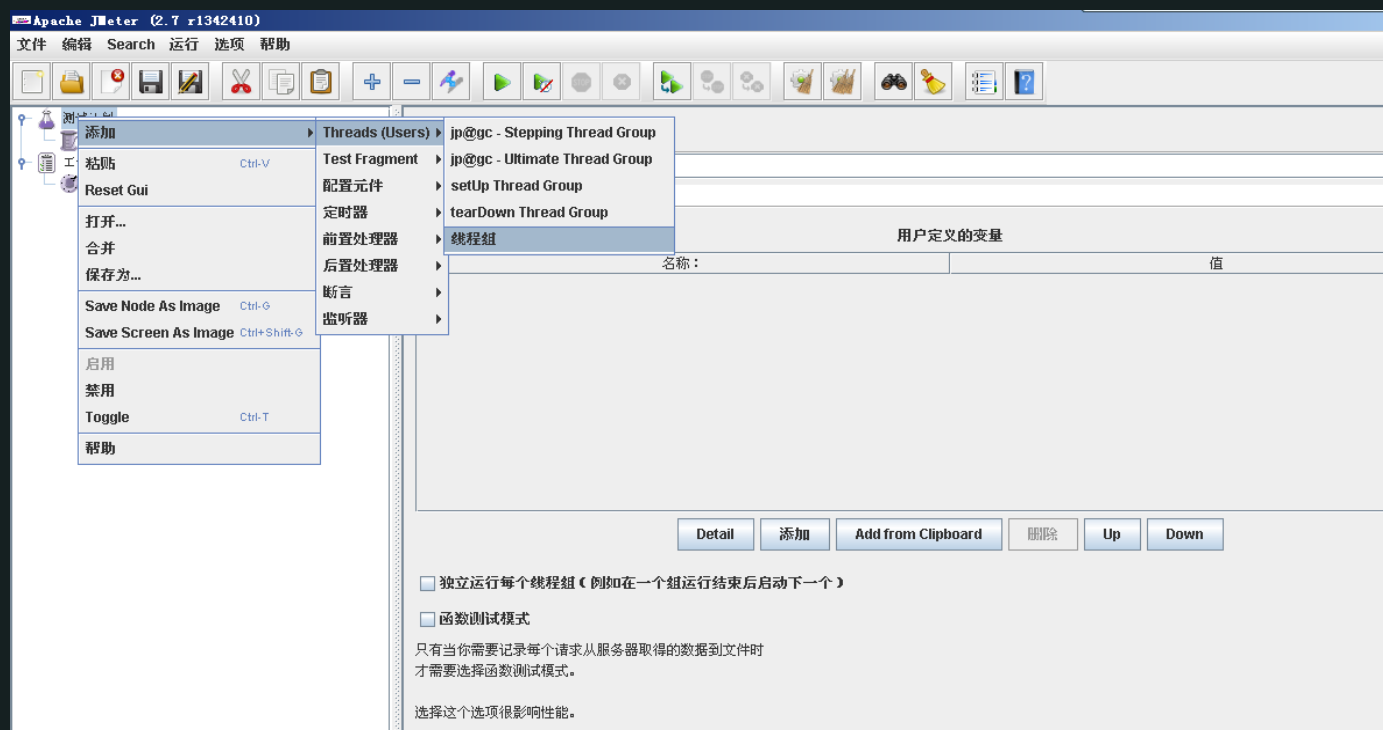
- 注意点：

- 使用BadBoy录制并导出成.jmx的脚本，导入到Jmeter后，需要修改step1的controller。



# Http代理服务器录制

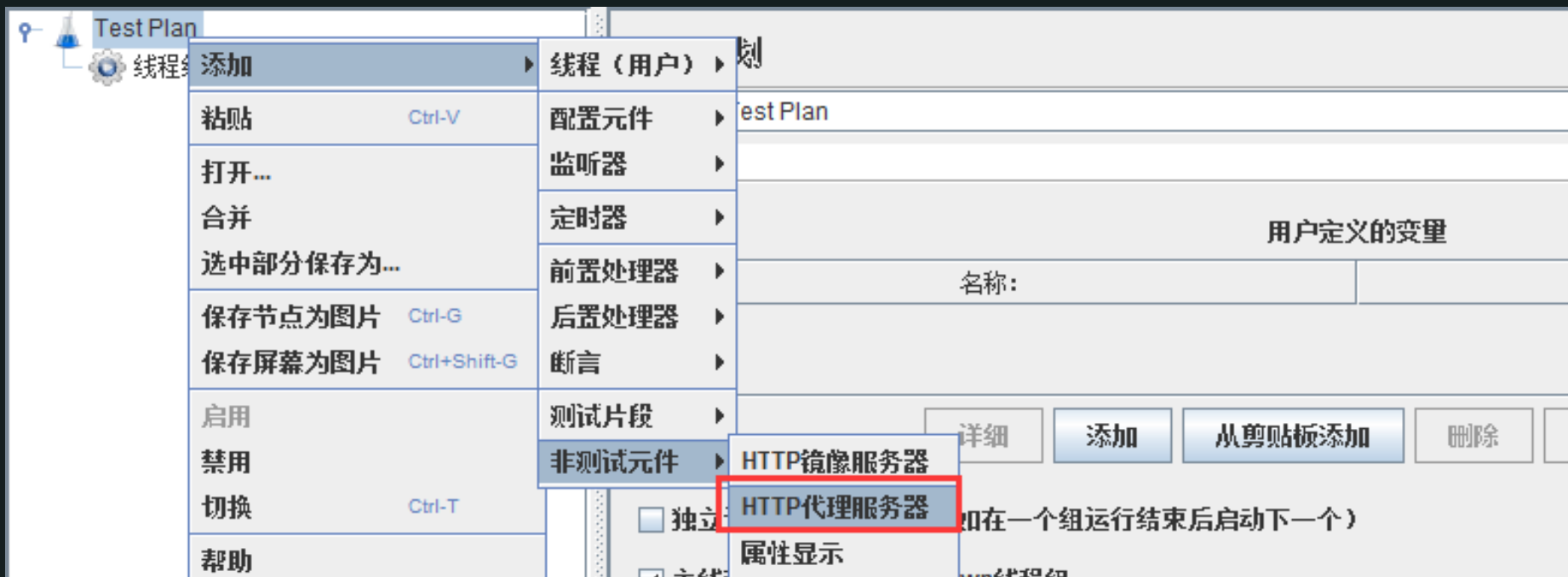
- Jmeter自带http proxy server，支持HTTP协议脚本录制。
- 第一步：启动Jmeter后，右键点击测试计划（Test Plan）  
添加 - Threads-Threads Group（线程组）





# Http代理服务器录制

- 第二步：右键点击测试计划-添加-非测试元件-HTTP代理服务器  
(HTTP Proxy Server)



# Http代理服务器录制

- 第三步：设置Http代理服务器

The screenshot shows the 'HTTP代理服务器' (HTTP Proxy Server) configuration window in JMeter. The left sidebar shows a tree structure with 'Test Plan' > '线程组' (Thread Group) > 'HTTP代理服务器' (HTTP Proxy Server) selected. The main panel is titled 'HTTP代理服务器' and contains the following sections:

- 名称:** HTTP代理服务器
- 注释:**
- State:** Includes buttons for '启动' (Start), '停止' (Stop), and '重启' (Restart).
- Global Settings:**
  - 端口:** 8888
  - HTTPS Domains:**
- Test Plan Creation** (selected tab) and **Requests Filtering** (unselected tab).
- Test plan content:**
  - 目标控制器:** Test Plan > 线程组
  - 分组:** 不对样本分组 (dropdown menu)
  - ☒ 记录HTTP信息头 ☐ 添加
- HTTP Sampler settings:**
  - Prefix:** (dropdown menu)
  - Create new transaction after request (ms):** (text input)
  - ☐ 从HTML文件获取所有内含的资源
  - ☐ 自动重定向
  - ☒ 使用 KeepAlive
  - ☒ 跟随重定向

# Http代理服务器的细节设置

- 名称：代理服务器的名字，默认即可。
- 端口：代理服务器的端口，默认是8888，如果被占用，可以换一个未被使用的端口
  - 在命令提示符里面输入`netstat -an`可以查看本机当前被使用的端口。
- 目标控制器：录制的脚本存放的位置，这里选择之前创建好的线程组
- 分组：对于录制的samplers是否分组，怎么控制分组
- 记录HTTP信息头：是否自动记录并生成http header信息。
- 添加断言：是否添加空白的断言步骤
- REgex matching：替换变量的时候是否使用正则表达式

# Http代理服务器的细节设置

- **Http sampler settings:** 指定http sampler的设置

- **Type:** 指定请求的模拟方式，默认是java的，可以选择httpclient4。

- 自动重定向和跟随重定向：

例如：A重定向到B，自动重定向在结果查看树中，只能看到B的调用及响应。

跟随重定向在结果查看树中，既能看到A的调用及响应，也能看到B的调用及响应

- **UseKeepAlive:** 在头文件里面添加KeepAlive属性。

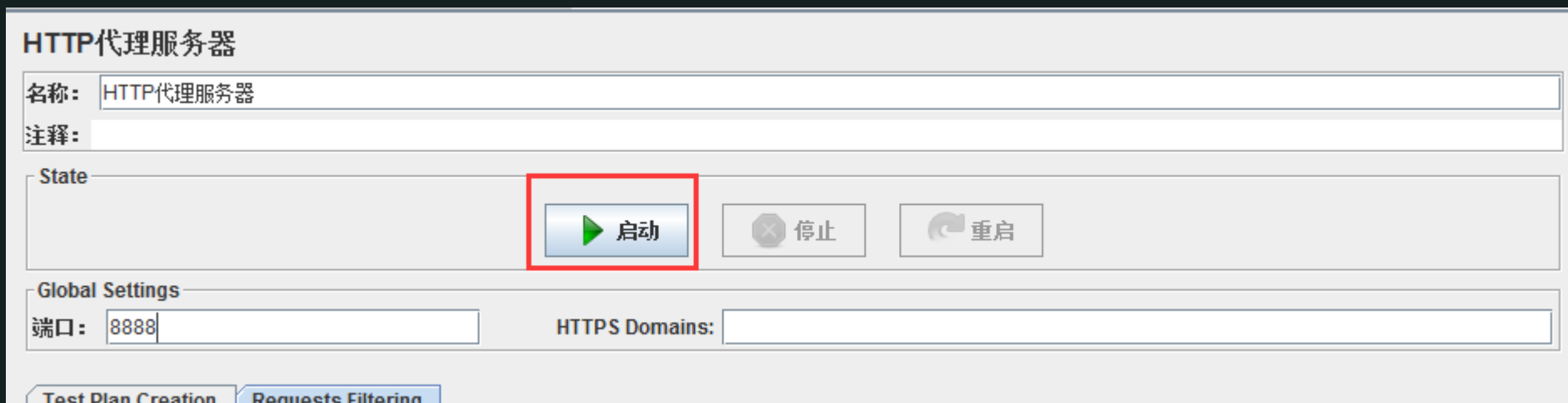
- **从HTML文件中获取所有内含的资源:** 获取除了html页面外所有内含的资源，包括图片等。一般不勾选。

# Http代理服务器的细节设置

- **Content-type filter:** 对Content-type进行过滤，多个类型之间使用逗号分隔。
  - 比如只想录制到text/html和text/xml格式的content，则在include里面输入 “text/html, text/xml”
- **包含/排除模式:**对请求的过滤，采用正则表达式的规则进行过滤。
  - 例如录制下来，有很多css, js, png等请求内容，想要过滤掉，可以在排除模式中添加： `.*\.css.*`

# Http代理服务器的细节设置

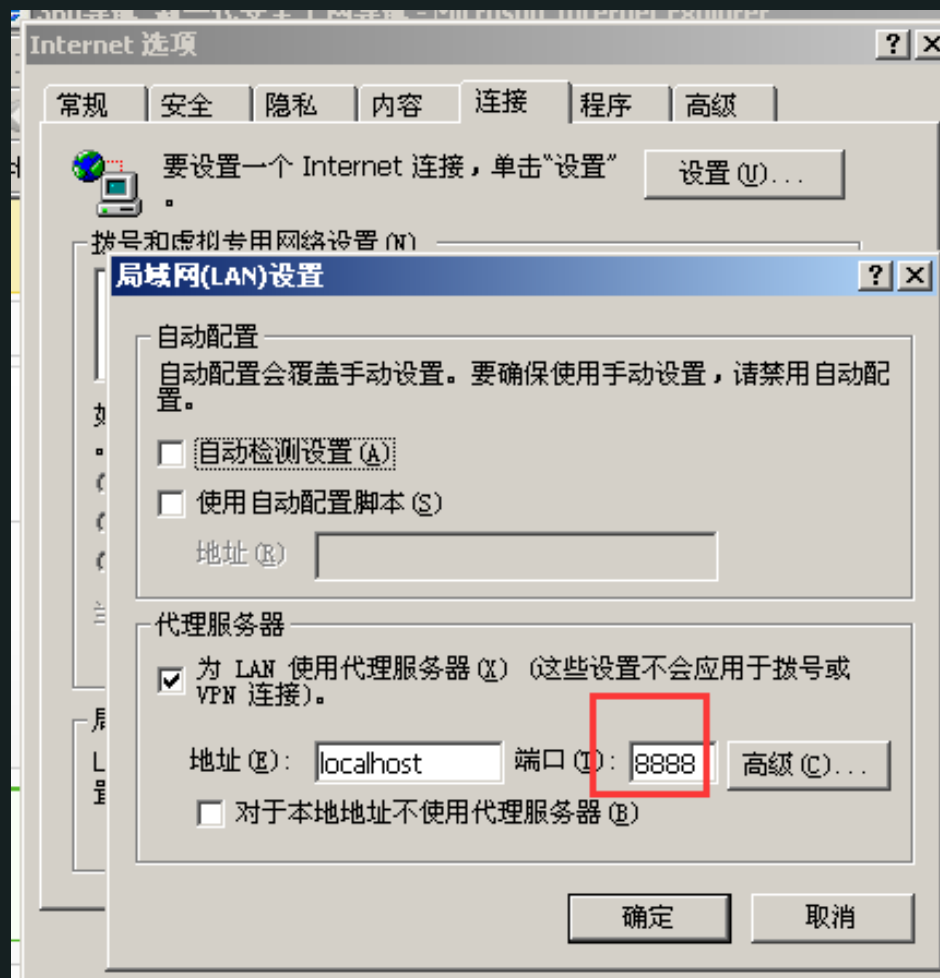
- 启动：启动http 代理服务。
  - 当弹出CA证书时，点击“确定”即可。
- 停止：停止http代理服务。
- 重启：重启http代理服务。



The screenshot shows the 'HTTP代理服务器' (HTTP Proxy Server) configuration window. It includes fields for '名称' (Name) and '注释' (Comment), both containing 'HTTP代理服务器'. Below these is a 'State' section with three buttons: '启动' (Start), '停止' (Stop), and '重启' (Restart). The '启动' button is highlighted with a red box. At the bottom, there are 'Global Settings' including '端口' (Port) set to '8888' and 'HTTPS Domains'. Navigation tabs at the bottom include 'Test Plan Creation' and 'Requests Filtering'.

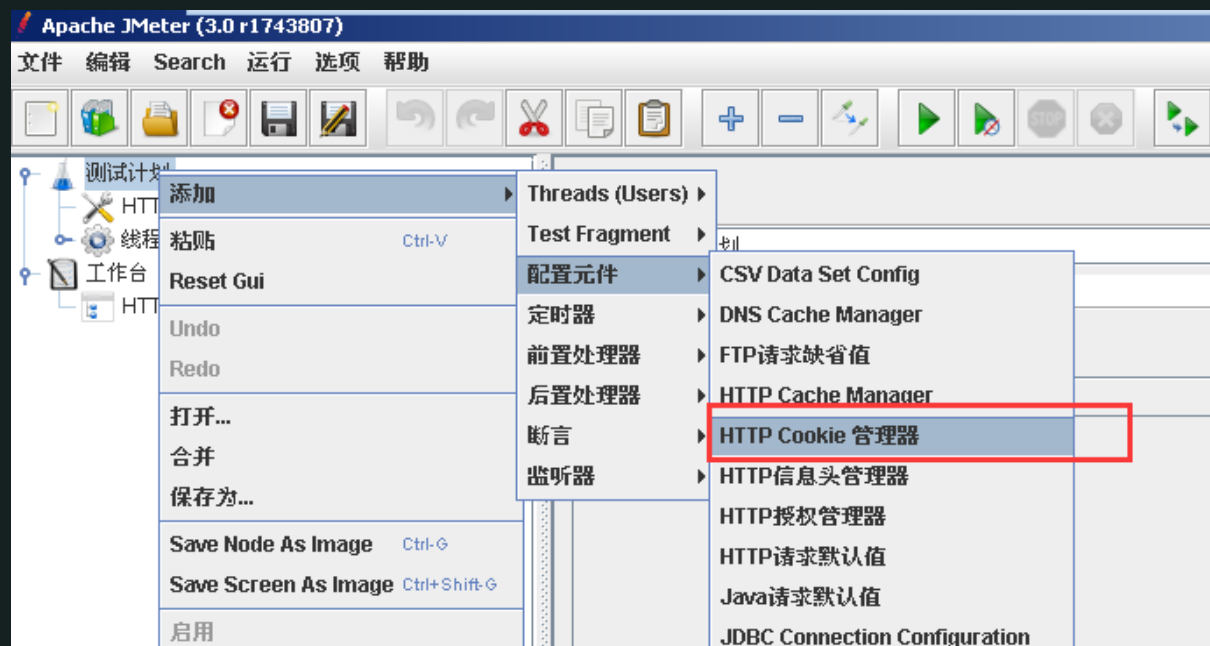
# Http代理服务器录制

- 第四步：浏览器设置
- 选中代理服务器：为LAN使用代理服务器。
- 配置地址为localhost或者127.0.0.1，端口为8888（端口号与Jmeter中http proxy server设置的端口保持一致）



# Http代理服务器录制

- 第五步:在浏览器中进行相应的业务操作，就可以看到在Jmeter的线程组下面新增了很多请求。
- 使用代理模式，最好添加上cookie管理器。HTTP Cookie Manager可以自动储存服务器发送给客户端的所有Cookie，并在发送请求时附加上合适的Cookie。





# 手动编写

- HTTP协议的系统我们可以通过录制的方式来生成原始脚本。
- 但是对于一些不能进行录制的系统，我们就需要根据对应的协议类型选择合适的取样器（sampler），手动编写对应的请求，实现模拟数据发送的效果。
- 演示手动编写
  - 线程组->取样器->察看结果树

# 课程目录

- JMeter 概要介绍
- JMeter 环境搭建
- JMeter 目录结构及常用元件
- JMeter 录制方法介绍
- **JMeter 参数化**
- JMeter事务和集合点
- JMeter 检查点
- JMeter 监听器
- 非GUI模式下运行JMeter
- JMeter案例与实践



# JMeter 参数化

## 三种参数化方式

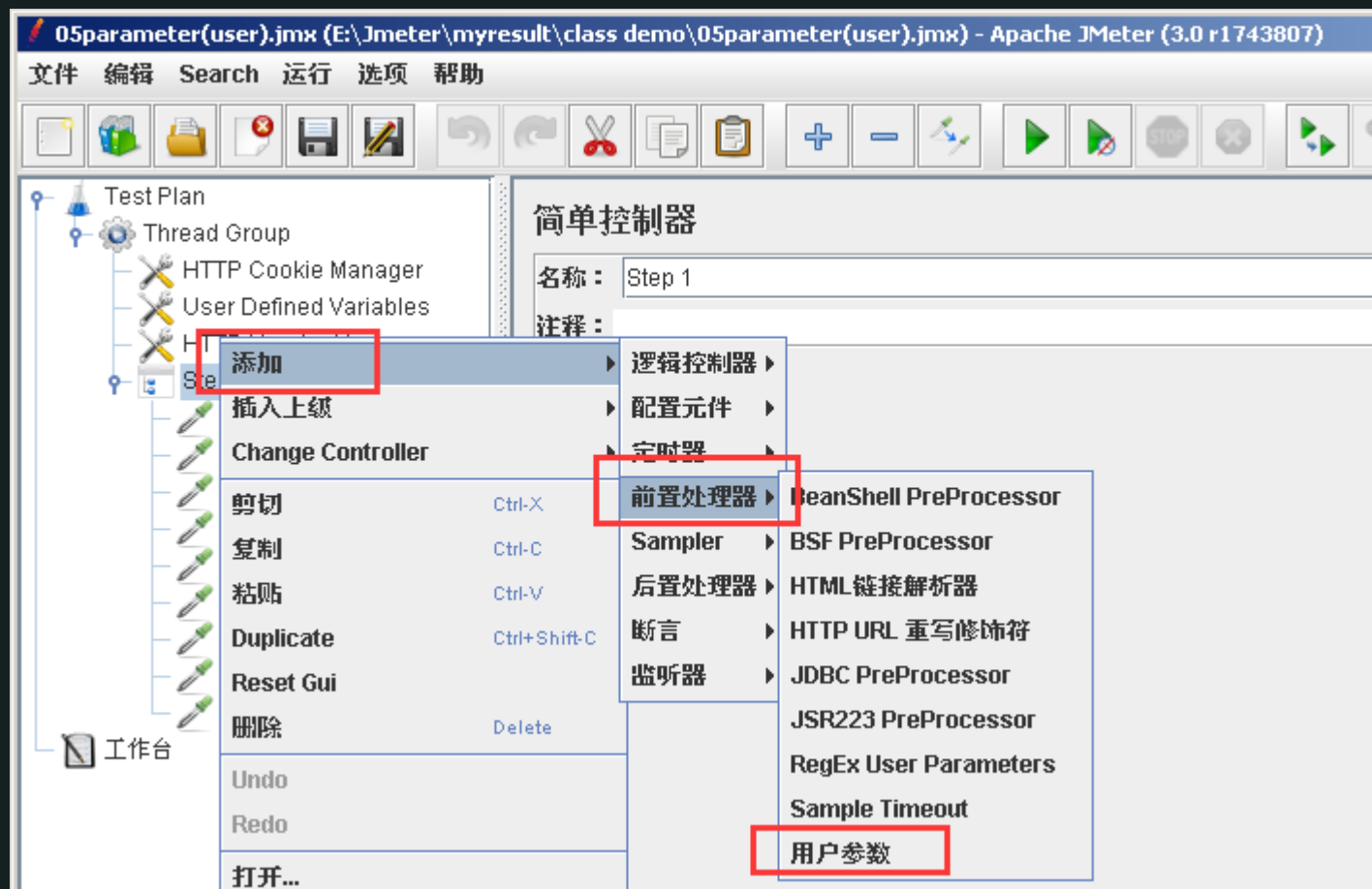
- 通过前置处理器参数化
- 通过CSV Data Set Config 参数化
- 借助函数助手方式采用随机参数化

# 通过前置处理器参数化

- 实例：论坛多用户登录。

- 步骤：

- ① Badboy录制论坛登陆的脚本
- ② 导入Jmeter
- ③ 添加-前置处理器 - 用户参数
- ④ 设置参数：添加变量和用户
- ⑤ 在脚本中用参数替代文本
- ⑥ 设置循环次数
- ⑦ 添加“察看结果树”并运行

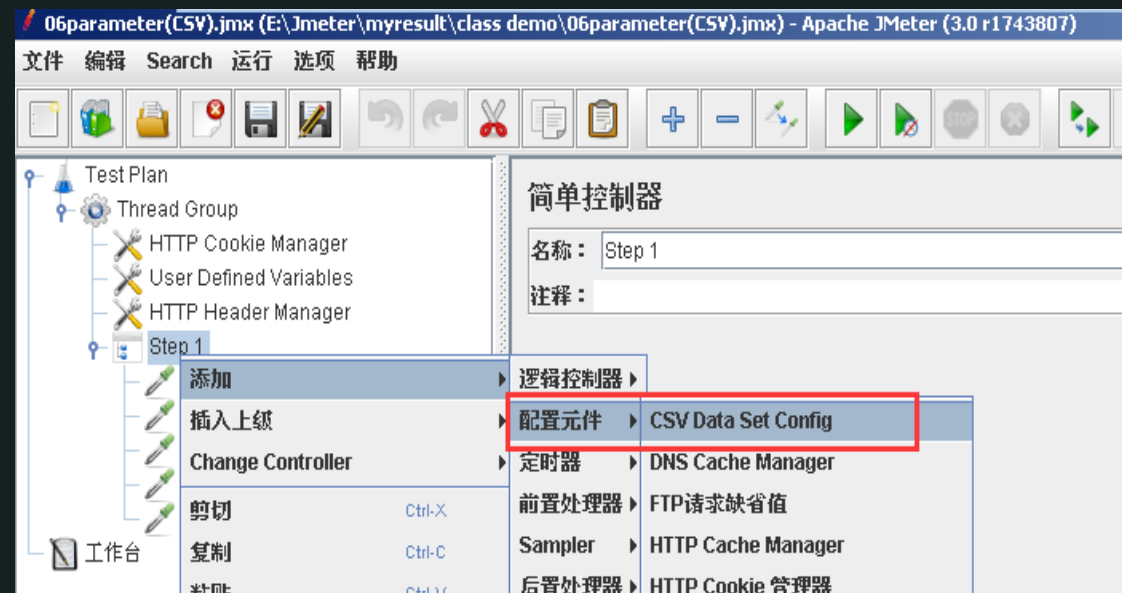
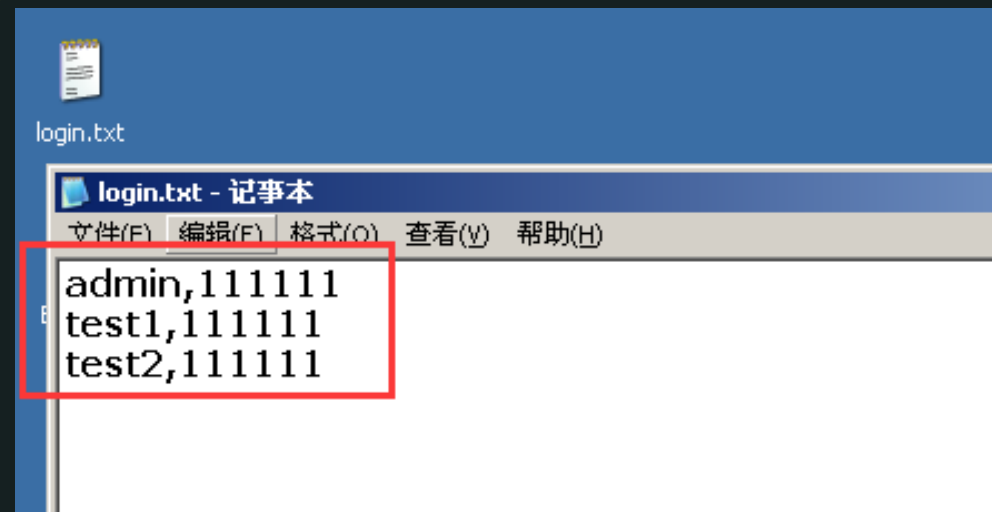


# 通过CSV Data Set Config 参数化

- 实例：论坛多用户登录。

- 步骤：

- ① Badboy录制论坛登陆的脚本
- ② 导入Jmeter
- ③ 制作含有三组登录名和密码的txt文件
- ④ 添加-配置元件-CSV Data Set config
- ⑤ 设置参数：添加变量和用户
- ⑥ 在脚本中用参数替代文本
- ⑦ 设置循环次数
- ⑧ 添加“察看结果树”并运行



# 借助函数助手方式采用随机参数化

- 实例：在论坛中使用随机内容发帖

- 步骤：

- ① Badboy录制论坛发帖的脚本
- ② 导入Jmeter
- ③ 选项-函数助手对话框
- ④ 设置RandomString
- ⑤ 生成函数字符串
- ⑥ 把论坛发帖的title替换成函数字符串
- ⑦ 设置循环次数。
- ⑧ 添加“察看结果树”并运行。



# 课程目录

- JMeter 概要介绍
- JMeter 环境搭建
- JMeter 目录结构及常用元件
- JMeter 录制方法介绍
- JMeter 参数化
- **JMeter事务和集合点**
- JMeter 检查点
- JMeter 监听器
- 非GUI模式下运行JMeter
- JMeter案例与实践



# 事务

- 作用：业务包含多个请求
- 步骤：
  - ① 添加-逻辑控制器-事务控制器
  - ② 添加-监听器-聚合报告
  - ③ 观察聚合报告





# 集合点

- 作用：“性能测试”理解为“**多用户**并发测试”，但真正的并发是不存在的，为了更真实的实现并发这概念，就需要在压力的地方设置集合点
- 步骤：
  - ① 添加-定时器-Synchronizing Timer
  - ② 设置集合点的人数
  - ③ 设置脚本循环次数
  - ④ 注意设置timeout的值



# 课程目录

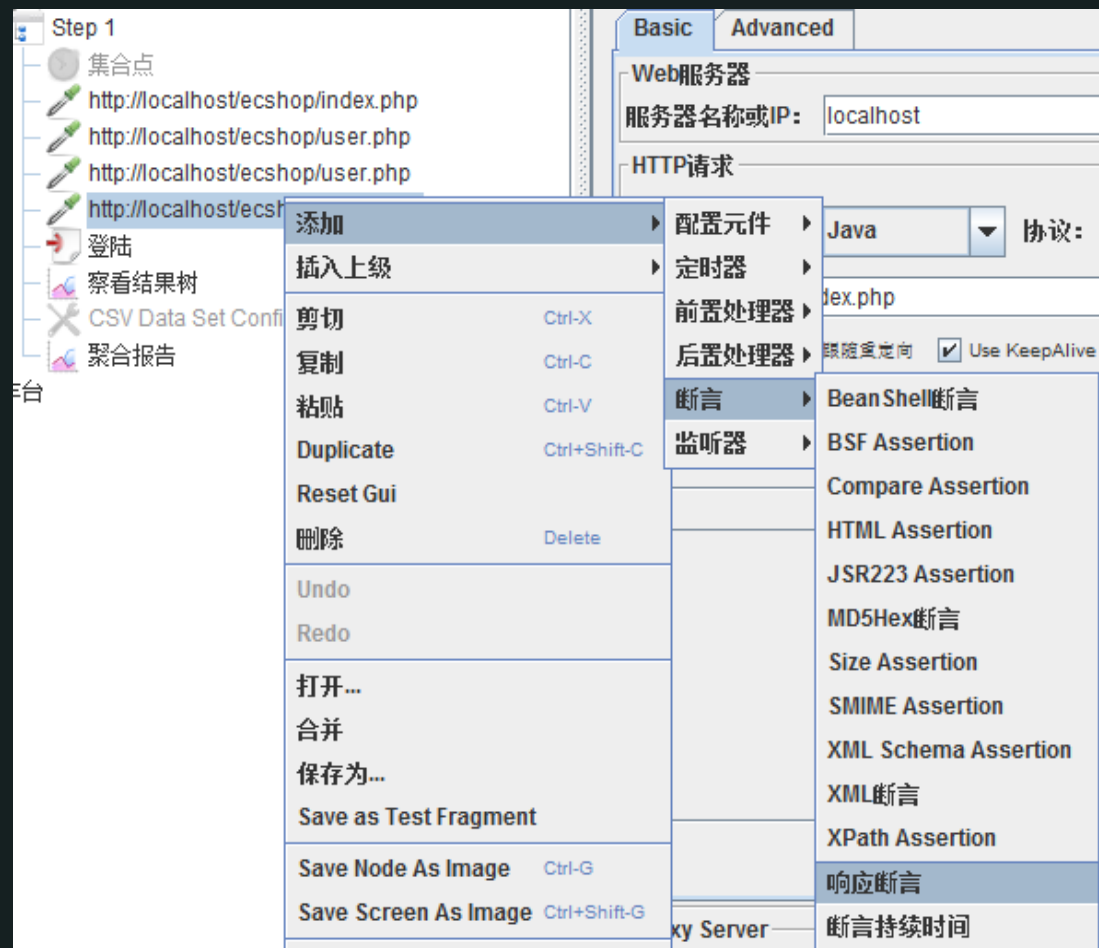
- JMeter 概要介绍
- JMeter 环境搭建
- JMeter 目录结构及常用元件
- JMeter 录制方法介绍
- JMeter 参数化
- JMeter事务和集合点
- **JMeter 检查点**
- JMeter 监听器
- 非GUI模式下运行JMeter
- JMeter案例与实践



# JMeter 检查点

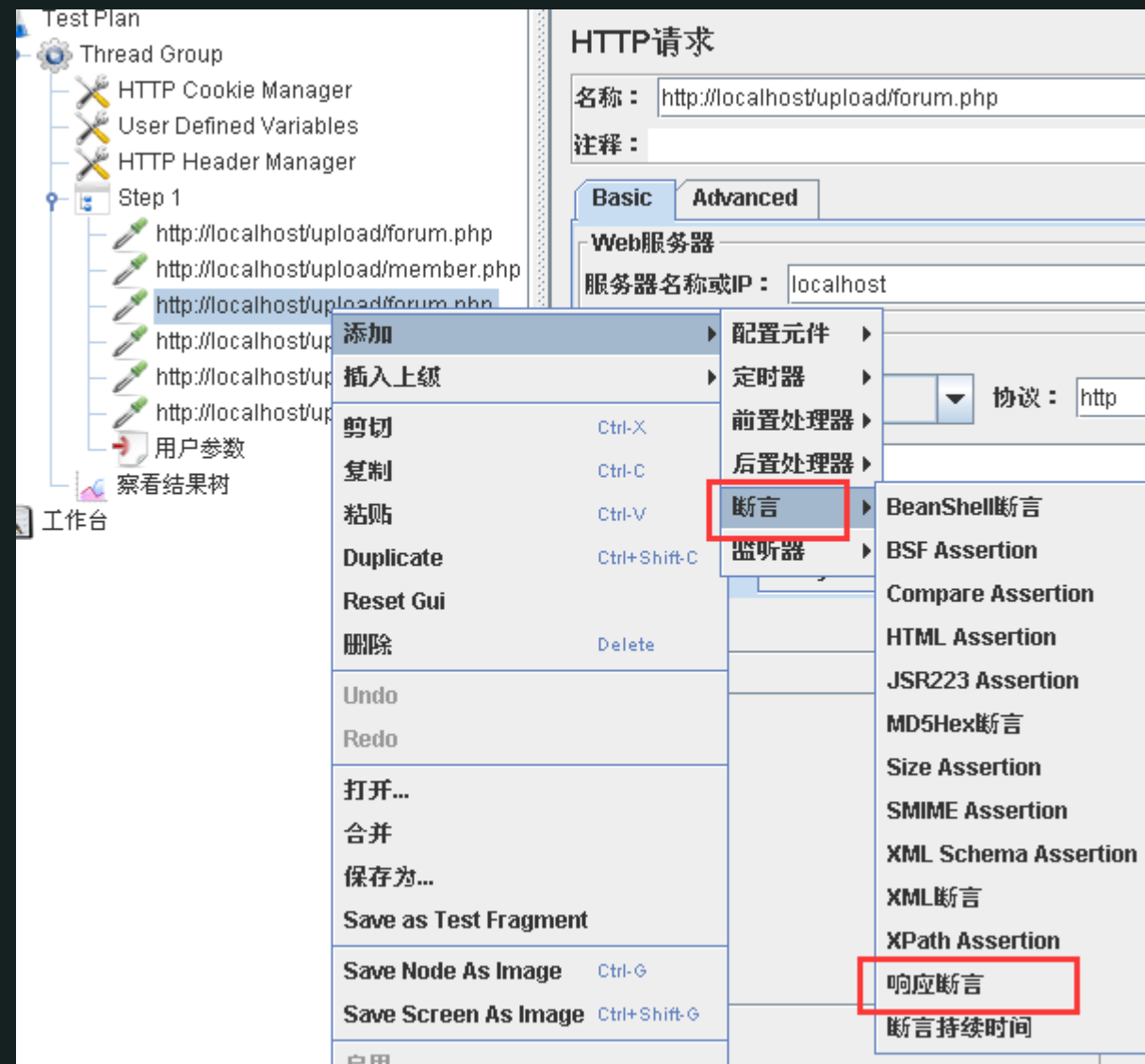
Jmeter中的常用检查点有：

- 内容检查断言
- 断言持续时间
- 断言结果大小



# 添加内容检查断言

- Jmeter中的检查点是通过添加断言来完成的。
- 实例：判断用户是否正确登陆论坛，检查用户登录后，有没有在论坛的首页出现用户名。
- 步骤：
  - ① 将前置处理器（用户参数）参数化的脚本另存一份。
  - ② 找到登陆后的页面请求，该页面包含了登陆成功后的用户名。
  - ③ 添加-断言-响应断言



# 添加内容检查断言

- 步骤:

④ 设置响应断言。

⑤ 添加断言的结果检查:

添加-监听器-断言结果

⑥ 查看断言结果: 观察对错表象。

响应断言

名称: 响应断言

注释:

Apply to:

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

要测试的响应字段

☒ 响应文本 ☐ Document (text) ☐ URL 样本 ☐ 响应代码 ☐ 响应信息 ☐ Response Header

模式匹配规则

☐ 包括 ☐ 匹配 ☐ Equals ☒ Substring ☐ 否

要测试的模式

要测试的模式



# 断言响应字段的含义

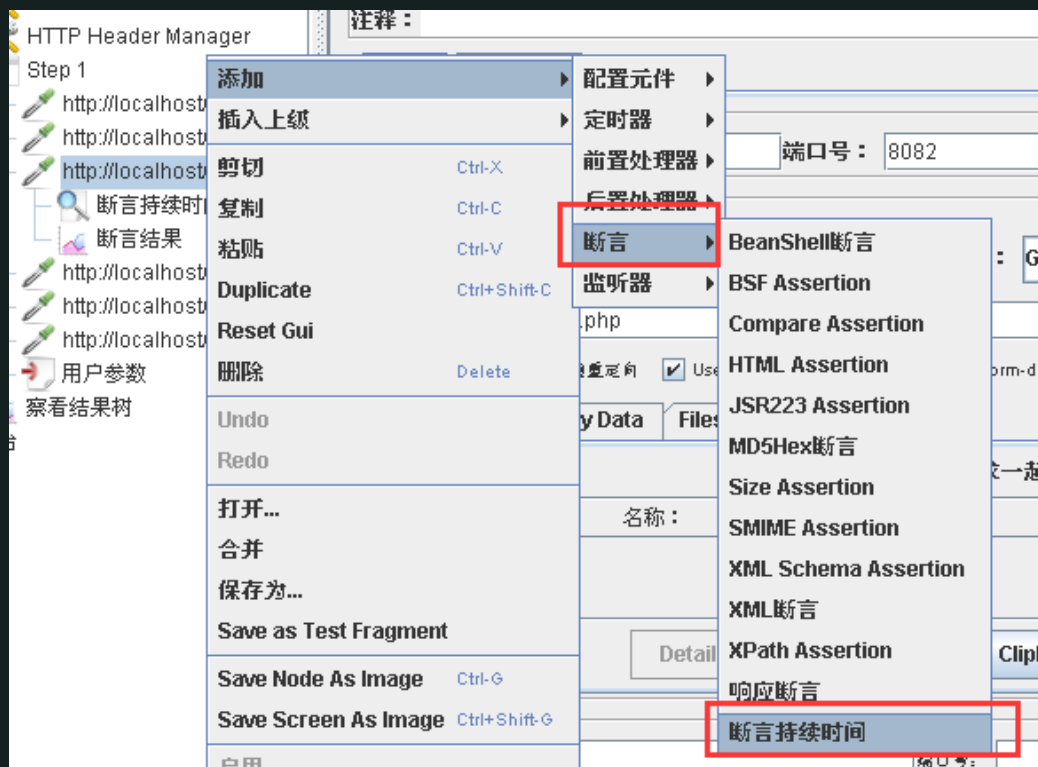
要测试的响应字段

☒ 响应文本 ☐ Document (text) ☐ URL 样本 ☐ 响应代码 ☐ 响应信息 ☐ Response Headers ☐ Ignore Status

- 响应文本: 服务器响应文本, 普通http响应, 勾选。
- Document(Text): Apache 解析的服务器响应内容, 若普通http请求, 不要选择这个。
- url样本: 对sample的url进行断言。如果请求没有重定向(302), 就是这个请求的url。如果有重定向(且跟随重定向), 那么url就包含请求url 和 重定向url。
- 响应代码: http响应代码, 如101,200,302,404,501等。但当我们要验证404,501等http响应代码时, 需要勾选 “ ignore status”。因为当http 响应代码为400,500时, jmeter默认这个请求时失败的。
- 响应信息: http响应代码对应的响应信息, 例如: OK, Found
- Response Header: 响应头部信息

# 断言持续时间（了解）

- 作用：检查允许的响应时间的最大值。
- 步骤：插入断言持续时间-插入断言结果-运行查看结果
- 含义：在持续时间中设置了1毫秒。从结果中可以看到要求的响应时间1毫秒，但是实际的响应时间为992毫秒，因此断言失败。



断言持续时间

名称：断言持续时间

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）：1

断言结果

名称：断言结果

注释：

所有数据写入一个文件

文件名  浏览... Log/Display Only: ☐ 仅日志

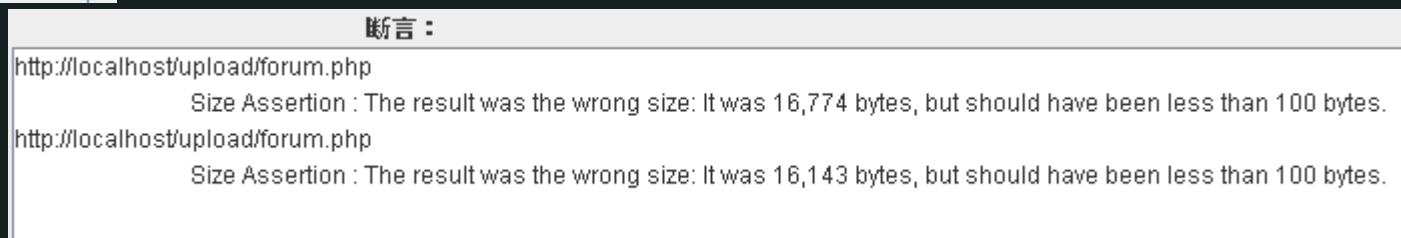
断言：

http://localhost/upload/forum.php  
断言持续时间：操作持续太长时间：他花费了992毫秒，但不应该超过1毫秒。

http://localhost/upload/forum.php  
断言持续时间：操作持续太长时间：他花费了1,090毫秒，但不应该超过1毫秒。

# 断言结果大小（了解）

- 作用：对于返回结果文件大小的标准定义，对响应结果字节大小的判断
- 步骤：插入Size Assertion-插入断言结果-运行查看结果
- 含义：在断言大小中，设置了小于100字节。从运行结果中可以看到实际的返回字节数是16774bytes，大于设置的100，因此断言失败。





# 课程目录

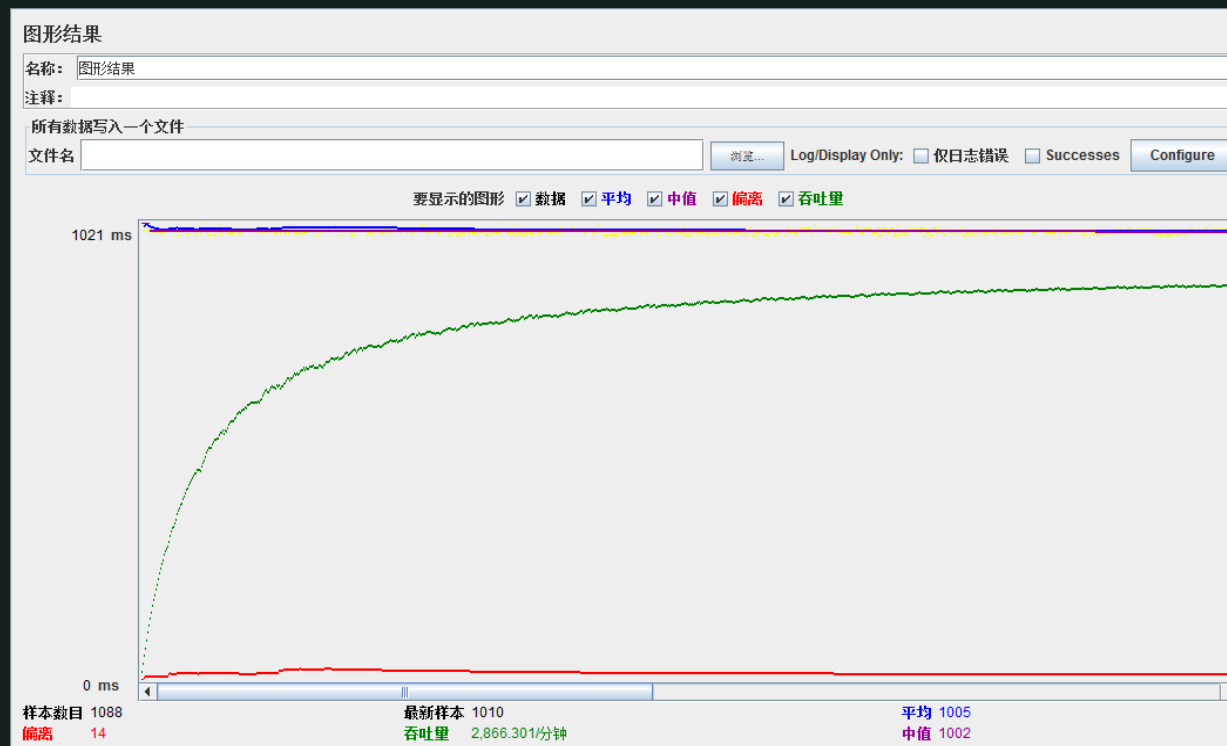
- JMeter 概要介绍
- JMeter 环境搭建
- JMeter 目录结构及常用元件
- JMeter 录制方法介绍
- JMeter 参数化
- JMeter事务和集合点
- JMeter 检查点
- **JMeter 监听器**
- 非GUI模式下运行JMeter
- JMeter案例与实践



# JMeter 常用监听器

常用监听器：

- 断言结果
- 图形结果
- 察看结果树
- 聚合报告
- Summary Report

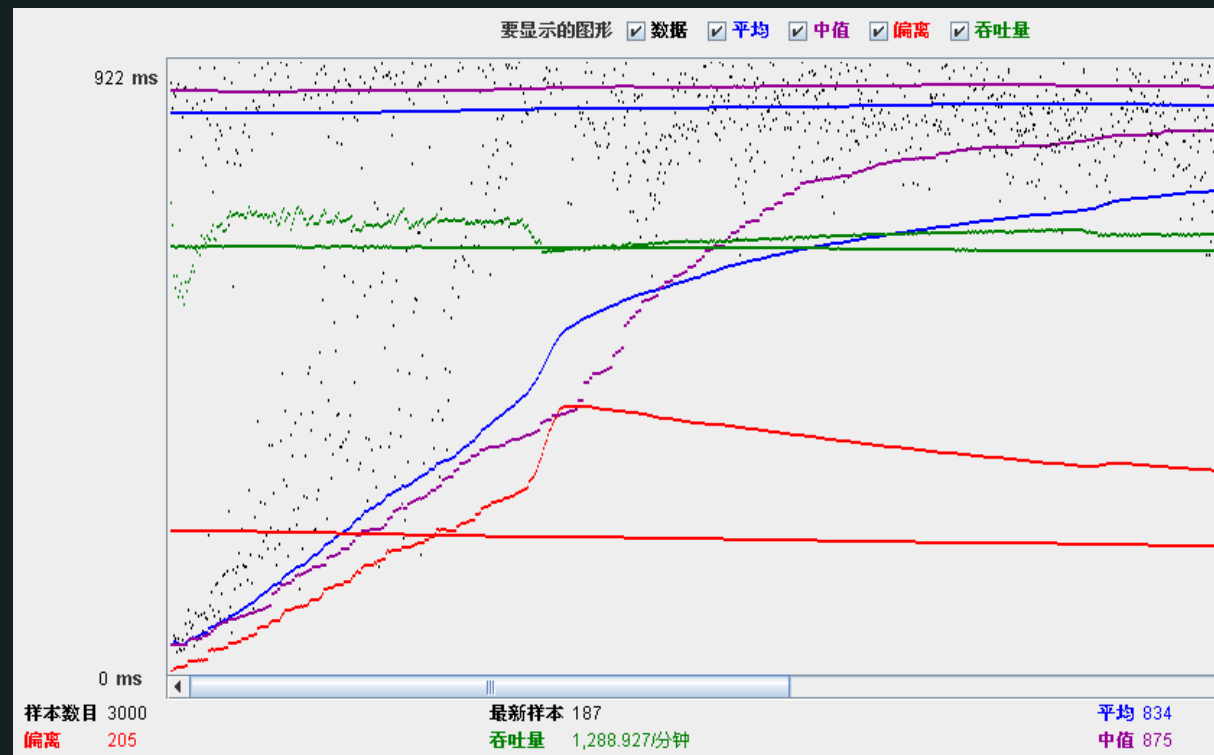


# 图形结果

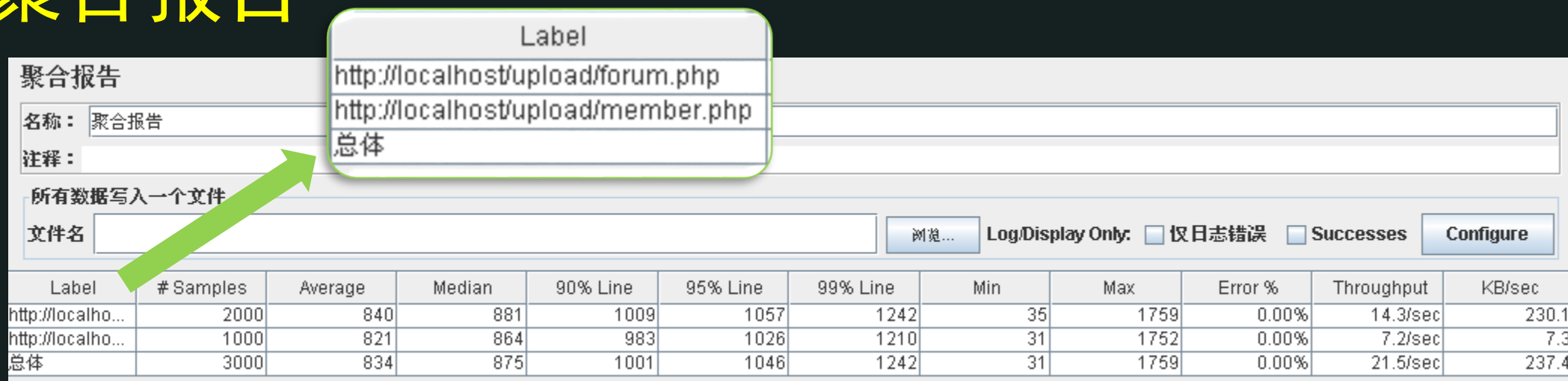
显示图线为随时间变化曲线，但X轴不是时间轴，是取样器个数的均匀分布轴

具体参数的含义：

- 样本数目：运行时得到的取样器响应结果个数
- 最新样本：最近一个取样器结果的响应时间
- 平均：所有取样器结果的响应时间平均值
- 偏离：所有取样器结果的响应时间标准差
- 吞吐量：每分钟响应的取样器结果个数
- 中值：所有取样器结果的响应时间中间值



# 聚合报告



The screenshot shows the 'Aggregating Report' (聚合报告) interface. A green arrow points to the 'Label' column header in the table below. The interface includes fields for 'Name' (名称: 聚合报告), 'Comment' (注释), and 'File Name' (文件名). It also has buttons for 'Browse...' (浏览...), 'Log/Display Only' (Log/Display Only), and 'Configure'.

Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Max	Error %	Throughput	KB/sec
http://localhost/upload/forum.php	2000	840	881	1009	1057	1242	35	1759	0.00%	14.3/sec	230.1
http://localhost/upload/member.php	1000	821	864	983	1026	1210	31	1752	0.00%	7.2/sec	7.3
总体	3000	834	875	1001	1046	1242	31	1759	0.00%	21.5/sec	237.4

含义如下:

- Label: 取样器名称
- Samples: 运行时得到的取样器响应结果个数
- Average: 所有取样器结果的响应时间平均值
- Median: 所有取样器结果的响应时间中间值
- 90%Line: 90%的用户低于或等于这个值。

单位: 毫秒

- Min: 所有取样器结果的响应时间最小值
- Max: 所有取样器结果的响应时间最大值
- Error%: 出错的取样器结果占有所有取样器结果的比例
- Throughput: 吞吐率, 每秒钟响应的取样器结果个数
- KB/sec: 每秒响应的数据流量

# Summary Report

Summary Report

名称：Summary Report

注释：

所有数据写入一个文件

文件名

浏览...

Log/Display Only: ☐ 仅日志错误 ☐ Successes

Configure

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes
http://localhost/...	2000	840	35	1759	206.66	0.00%	14.3/sec	230.11	16452.9
http://localhost/...	1000	821	31	1752	203.52	0.00%	7.2/sec	7.27	1038.6
总体	3000	834	31	1759	205.83	0.00%	21.5/sec	237.37	11314.8

含义如下：

- Std.Dev.: 所有取样器结果的响应时间标准差
- KB/sec: 每秒钟响应的数据流量
- Avg.Bytes: 所有取样器返回http response data字节数的平均值

# 课程目录

- JMeter 概要介绍
- JMeter 环境搭建
- JMeter 目录结构及常用元件
- JMeter 录制方法介绍
- JMeter 参数化
- JMeter事务和集合点
- JMeter 检查点
- JMeter 监听器
- 非GUI模式下运行JMeter
- JMeter案例与实践



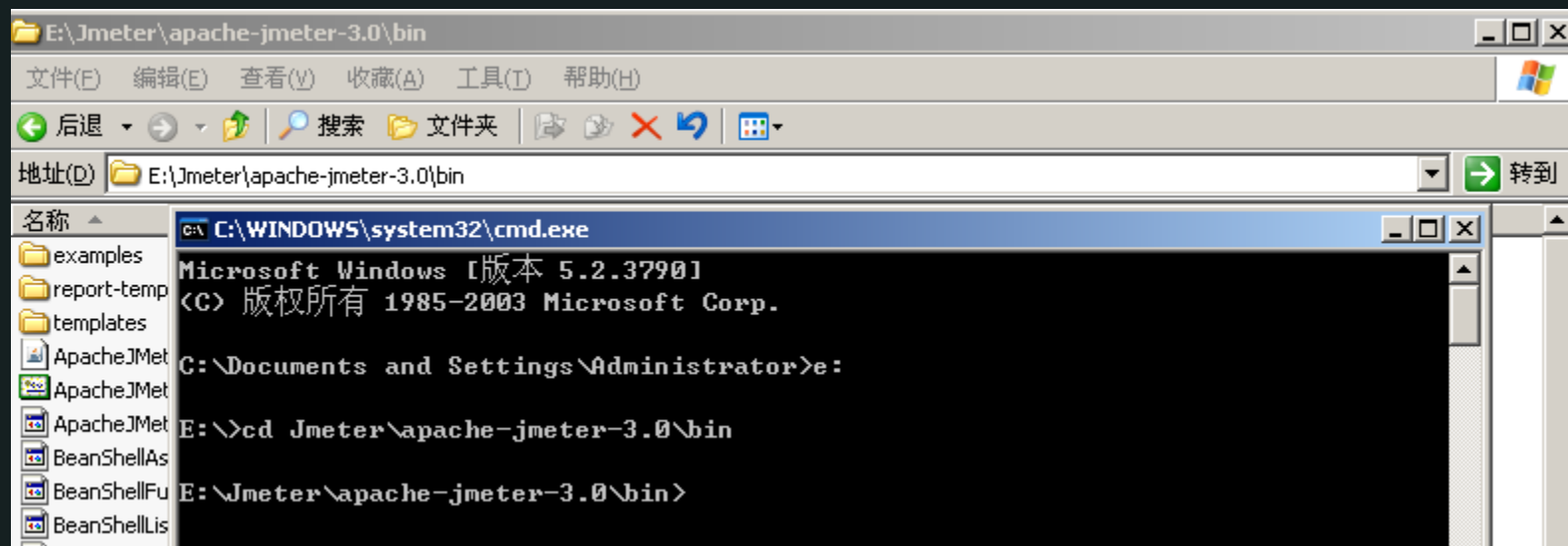
# 非GUI模式下运行JMeter

用途：

- ① GUI模式下运行非常消耗资源，所以可以使用命令行模式运行JMeter测试脚本
- ② 命令行下可以批量运行脚本，达到无人值守的目的。

步骤：

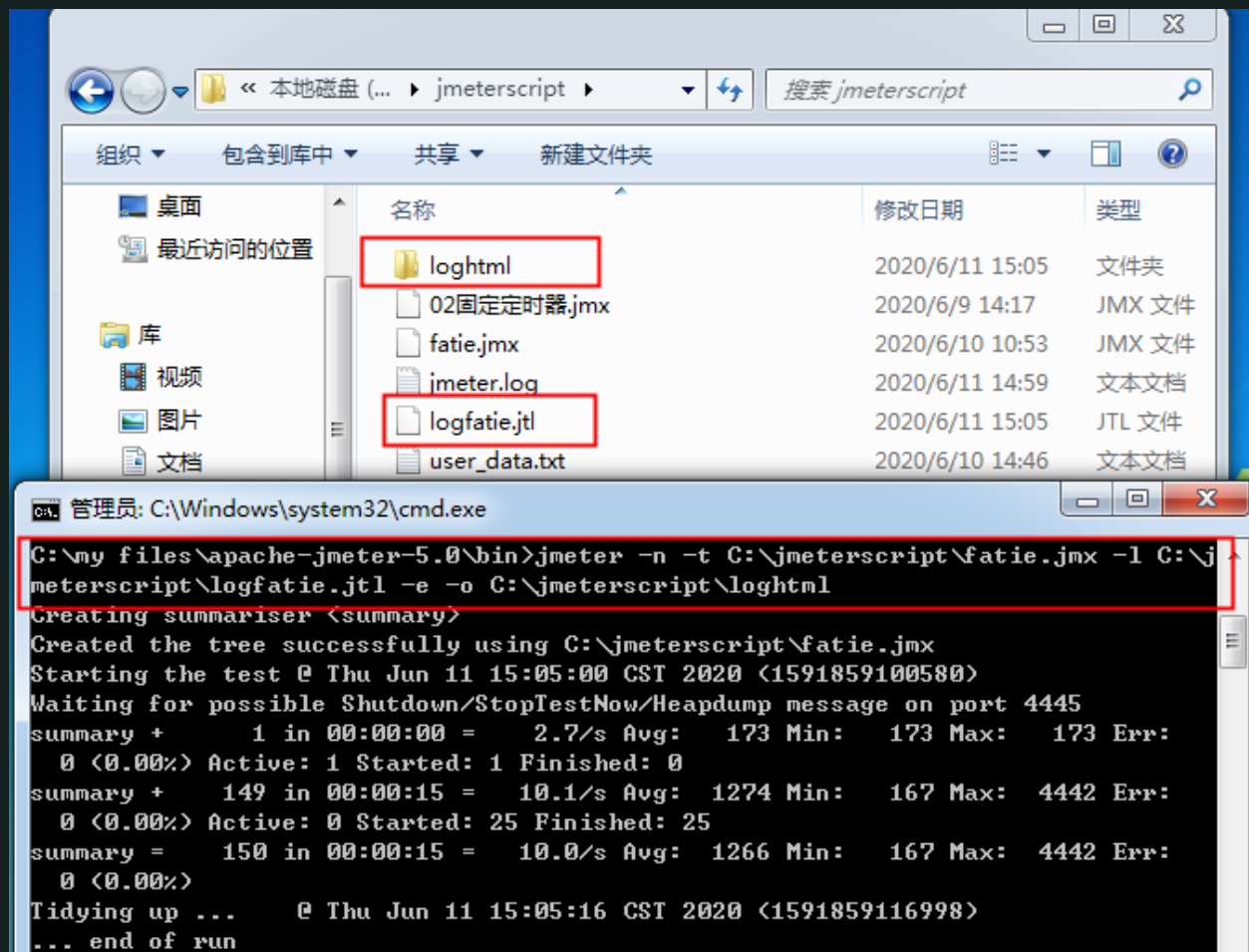
- ① 开始-运行-cmd-通过cd命令进入JMeter安装的bin目录



# 非GUI模式下运行JMeter

## ② 在bin目录下运行

```
jmeter -n -t E:\Jmeter\myresult\test\fatie.jmx -l E:\Jmeter\myresult\test\fatie.jtl -e -o loghtml
```

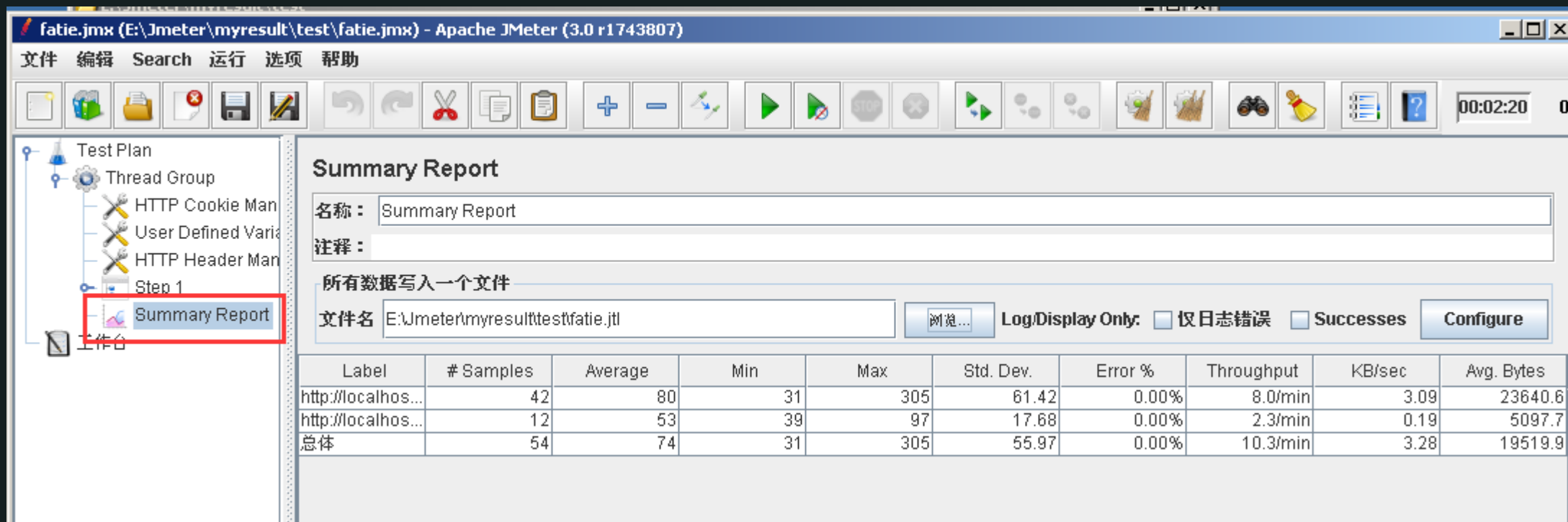


- -n 表示non-GUI;
- -t 指定要运行的jmeter脚本文件
- -l 记录采样器Log的文件，运行后会自动生成, 此文件必须不存在
- -e 在脚本运行结束后生成html报告
- -o 保存html报告的地址, 此文件必须不存在



# 非GUI模式下运行JMeter

- ③ 通过JMeter图形界面打开脚本，添加监听器，如summary report，在summary report通过“浏览”打开.jtl文件，可以看到相应的测试运行结果分析

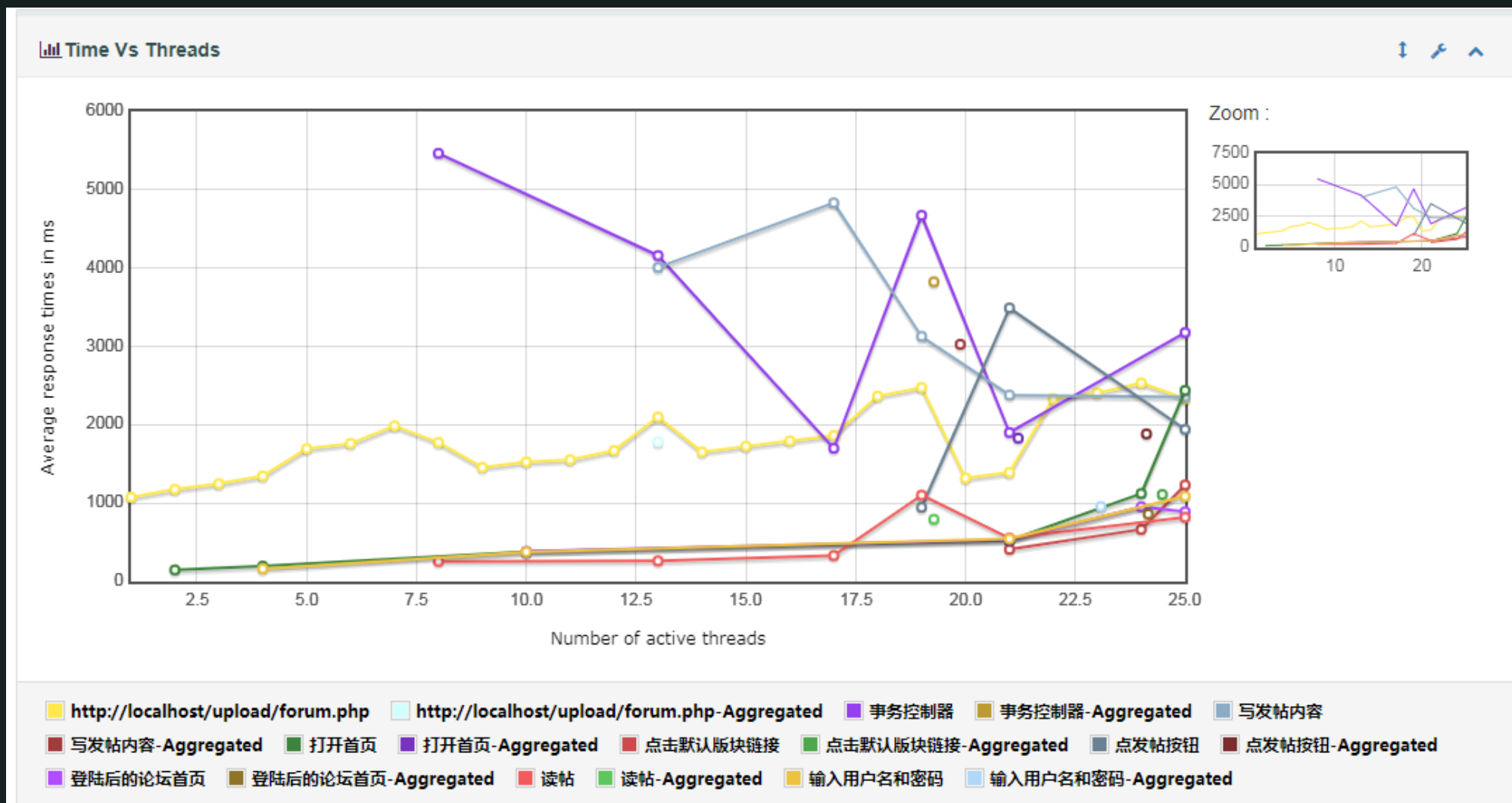


The screenshot shows the Apache JMeter 3.0 GUI. The title bar indicates the file path: `fatie.jmx (E:\Jmeter\myresult\test\fatie.jmx) - Apache JMeter (3.0 r1743807)`. The menu bar includes: 文件, 编辑, Search, 运行, 选项, 帮助. The toolbar contains various icons for file operations and test execution. The left sidebar shows the test plan structure: Test Plan, Thread Group, HTTP Cookie Manager, User Defined Variables, HTTP Header Manager, Step 1, and Summary Report (highlighted with a red box). The main panel displays the Summary Report configuration. The name is "Summary Report". The comment field is empty. The option "所有数据写入一个文件" (All data written to one file) is selected. The file name is `E:\Jmeter\myresult\test\fatie.jtl`. There are buttons for "浏览..." (Browse...), "Log/Display Only:", "仅日志错误" (Log only errors), "Successes", and "Configure". Below the configuration, a table displays the test results.

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes
http://localhos...	42	80	31	305	61.42	0.00%	8.0/min	3.09	23640.6
http://localhos...	12	53	39	97	17.68	0.00%	2.3/min	0.19	5097.7
总体	54	74	31	305	55.97	0.00%	10.3/min	3.28	19519.9

# 非GUI模式下运行JMeter

- html报告的Demo



# 课程目录

- JMeter 概要介绍
- JMeter 环境搭建
- JMeter 目录结构及常用元件
- JMeter 录制方法介绍
- JMeter 参数化
- JMeter事务和集合点
- JMeter 检查点
- JMeter 监听器
- 非GUI模式下运行JMeter
- JMeter案例与实践



# 创建SOAP WebService 测试计划

# 创建SOAP WebService 测试计划

- Web Service是由企业发布的完成其特定商务需求的在线应用服务，其他公司或应用软件能够通过Internet来访问并使用这项在线服务。
- WebService是一个应用组件。各应用程序通过网络协议和规定的一些标准数据格式（HTTP，XML，SOAP）来访问Web Service，通过Web Service内部执行得到所需结果。
- SOAP：简单对象访问协议。是交换数据的一种协议规范，是一种轻量的、简单的、基于XML的协议

# 创建SOAP Webservice 测试计划

实例：JMeter通过发送包含SOAP信息的HTTP请求进行WebService  
压力测试

场景：5个并发用户\*1请求\*重复2遍=10个HTTP请求；

监控：响应后返回的信息，图形结果，聚合报告

# 创建SOAP WebService 测试计划

操作步骤：

① Web Services地址：

[http://www.webxml.com.cn/zh\\_cn/web\\_services.aspx](http://www.webxml.com.cn/zh_cn/web_services.aspx)

② 学生熟悉Web Services站点的内容和作用

③ 打开Jmeter，新建一个测试计划。

④ 为测试计划添加HTTP信息头管理器，并在界面添加Content-Type信息

Content-Type	text/xml; charset=utf-8
--------------	-------------------------

⑤ 添加线程组，线程属性中完成场景要求的压力值。

⑥ 添加Sampler-http请求。完成服务器、路径、BodyData的设置。

⑦ 添加察看结果树、图形结果、聚合报告

# 创建JDBC测试计划



# 使用Jmeter对mysql进行性能测试

- 准备工作：
  - 在测试计划中添加Mysql的驱动包
  - 本机的mysql的用户名和密码
  - Mysql中的数据库，本实例中，在mysql中新建了一个learn的数据库，配有一张userinfo的表，任意添加两行数据。

# 使用Jmeter对mysql进行性能测试

- 步骤：

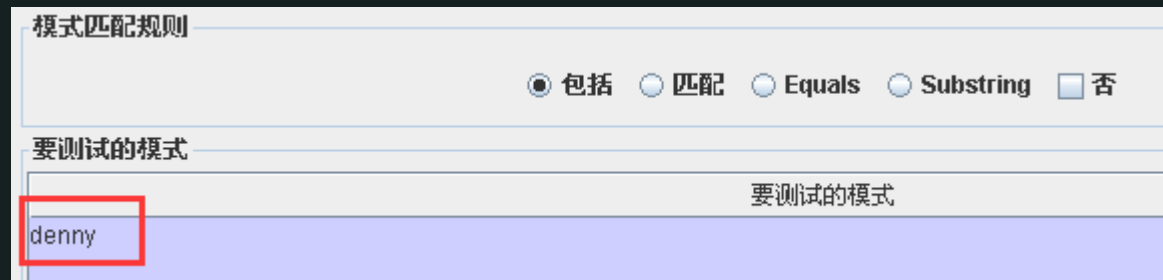
- ① 打开Jmeter，创建线程组，性能需求为5个并发用户\*1请求\*重复10遍=50 JDBC Request请求；
- ② 线程组-添加-配置元件-JDBC Connection Configuration
- ③ 线程组-添加-Sampler-JDBC Request

The screenshot shows the 'JDBC Request' configuration window. The 'Name' field is 'JDBC Request'. The 'Variable Name Bound to Pool' section has 'Variable Name' set to 'mysql01', with a red box around it and a red note '配置中定义的变量名'. The 'SQL Query' section has 'Query Type' set to 'Select Statement' and the query text 'select \* from userinfo;' in the text area, which is also highlighted with a red box and a red note '请求中执行的语句'.

The screenshot shows the 'JDBC Connection Configuration' window. The 'Name' field is 'JDBC Connection Configuration'. The 'Variable Name Bound to Pool' section has 'Variable Name' set to 'mysql01', with a red box around it and a red note '变量名在JDBC Request中使用'. The 'Connection Pool Configuration' section has 'Max Number of Connections' set to 10, 'Max Wait (ms)' set to 10000, 'Time Between Eviction Runs (ms)' set to 60000, 'Auto Commit' set to 'True', and 'Transaction Isolation' set to 'DEFAULT'. The 'Connection Validation by Pool' section has 'Test While Idle' set to 'True', 'Soft Min Evictable Idle Time(ms)' set to 5000, and 'Validation Query' set to 'Select 1'. The 'Database Connection Configuration' section has 'Database URL' set to 'jdbc:mysql://localhost:3306/learn', 'JDBC Driver class' set to 'com.mysql.jdbc.Driver', 'Username' set to 'root', and 'Password' is empty.

# 使用Jmeter对mysql进行性能测试

④ 设置断言检查返回结果，检查是否在返回中有danny的数据



⑤ 添加监听器：察看结果树、图形结果、聚合报告、断言结果

# Any Questions?



*THANK YOU!*