JMeter 基本使用方法

白羽

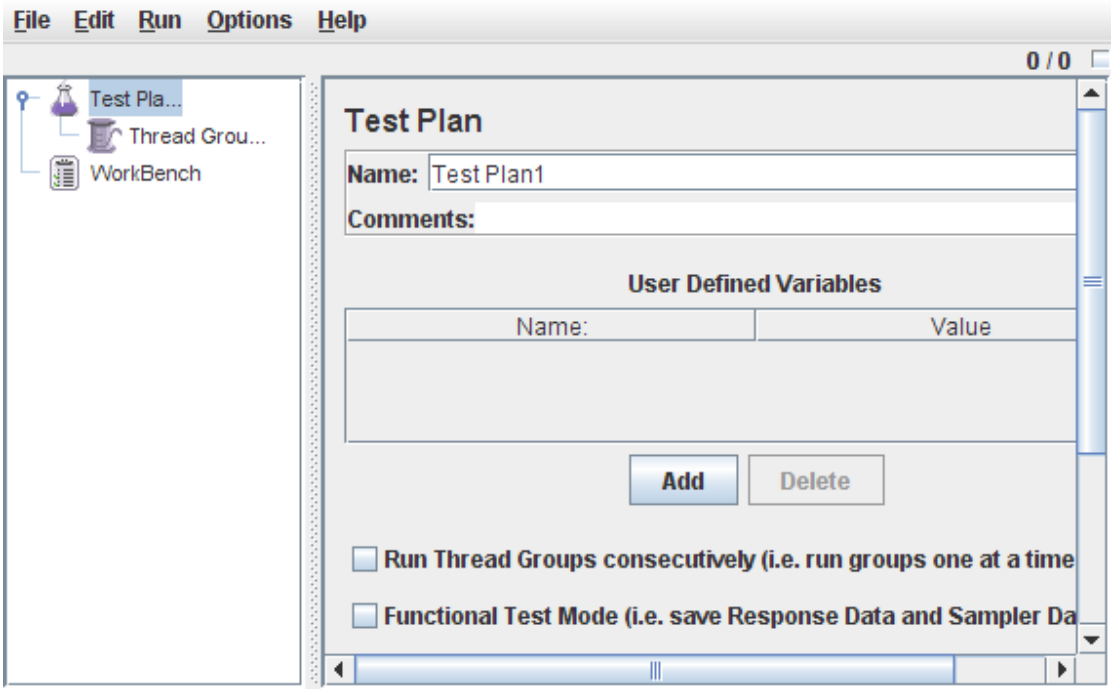
1 环境搭建

此部分是我借用 LoadRunner 的 Sample 程序作为服务器，通过 JMeter 进行 Web 性能测试的过程。希望对学习 JMeter 的朋友有所帮助。

2 使用 JMeter 录制脚本

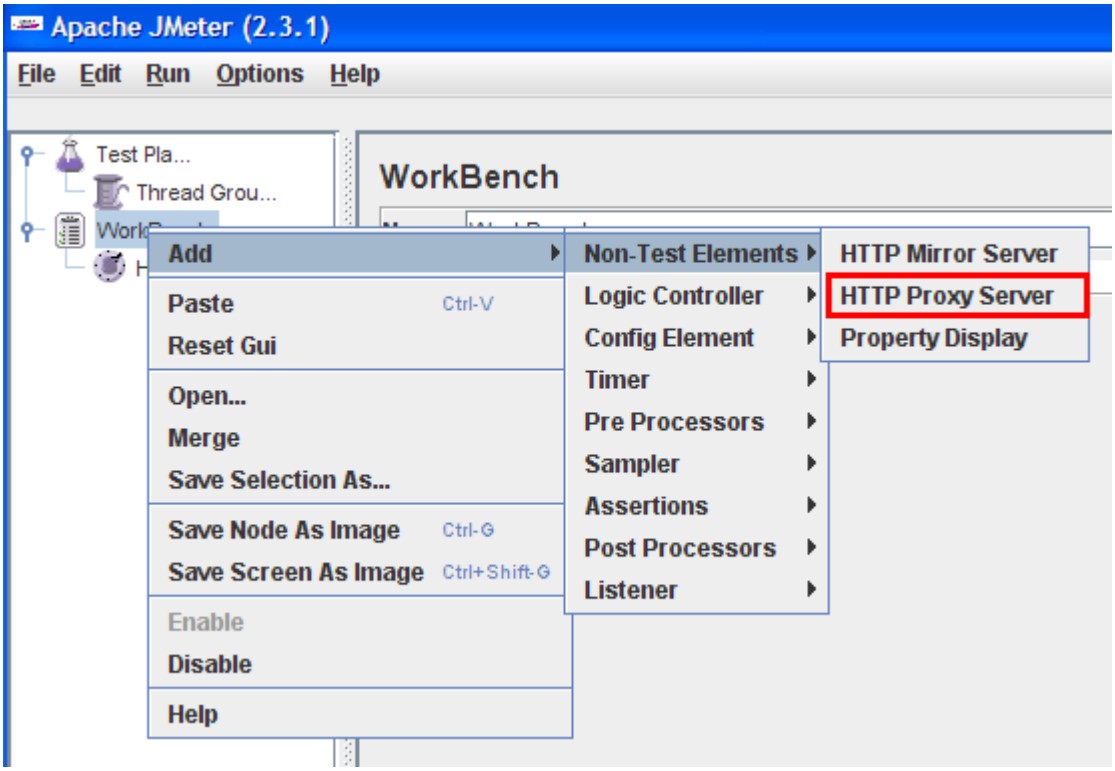
2.1 建立 JMeter 测试计划(Test Plan)

打开 JMeter，看到左边显示一个空的测试计划，把该测试计划改名为 TestPlan1。右键单击该测试计划，选择“添加(Add)”-“线程组(Thread Group)”，添加一个线程组，改名为 TestGroup1。如下图所示：

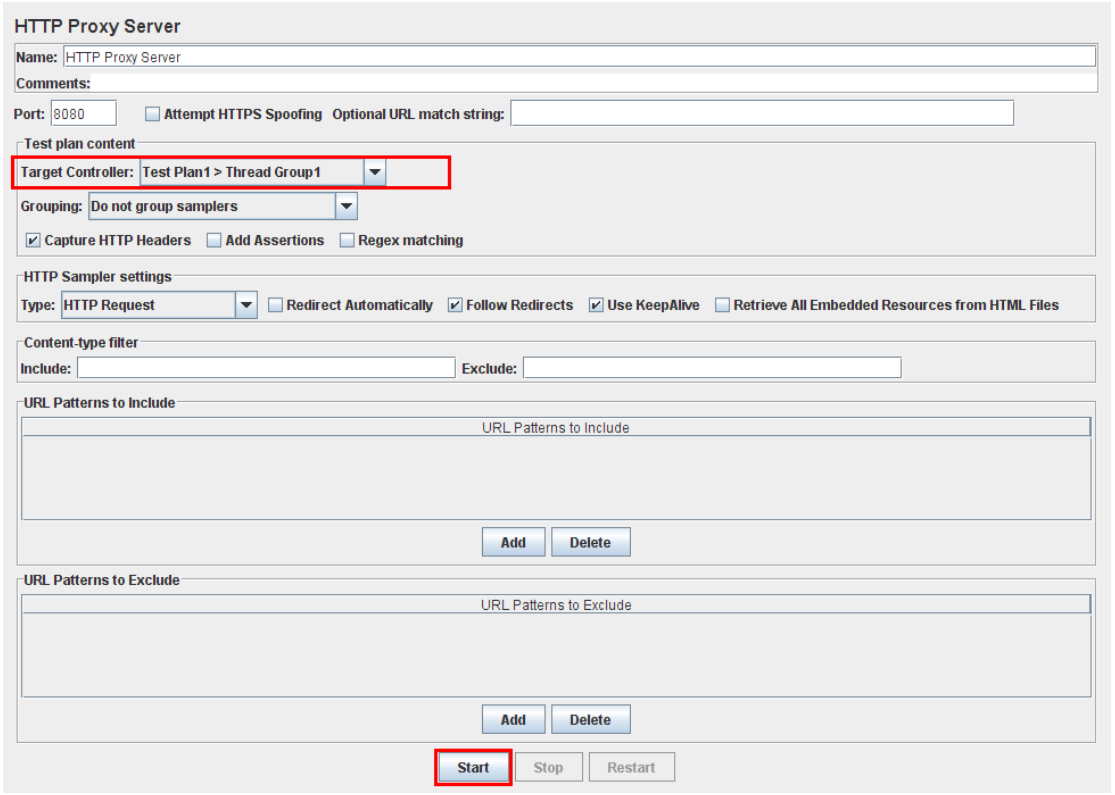


2.2 设置并启动 JMeter 代理服务器

右键单击在“工作台(Work Bench)”，选择“非测试元件(Non-Test Elements)”中的“Http 代理服务器(HTTP Proxy Server)”，如下图所示：



设置该“Http 代理服务器”(HTTP Proxy Server)，“目标控制器(Target Controller)”选择刚才建立的线程组(Test Plan1>Thread Group1)。

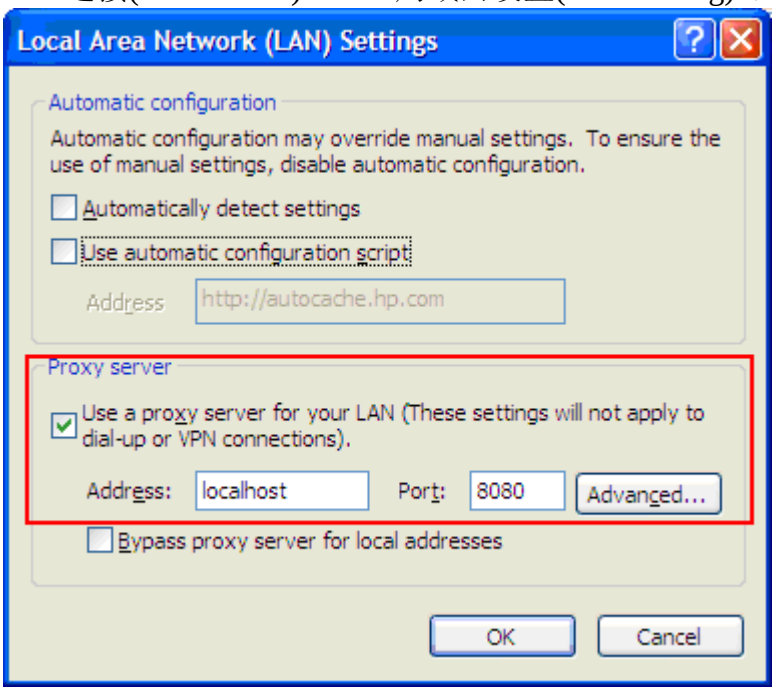


点击“启动”，启动该代理服务器。

2.3 设置 IE 的代理服务器配置

测试工程师打开 IE 界面，选择“工具(Tools)”->“Internet 选项(Internet Option)”

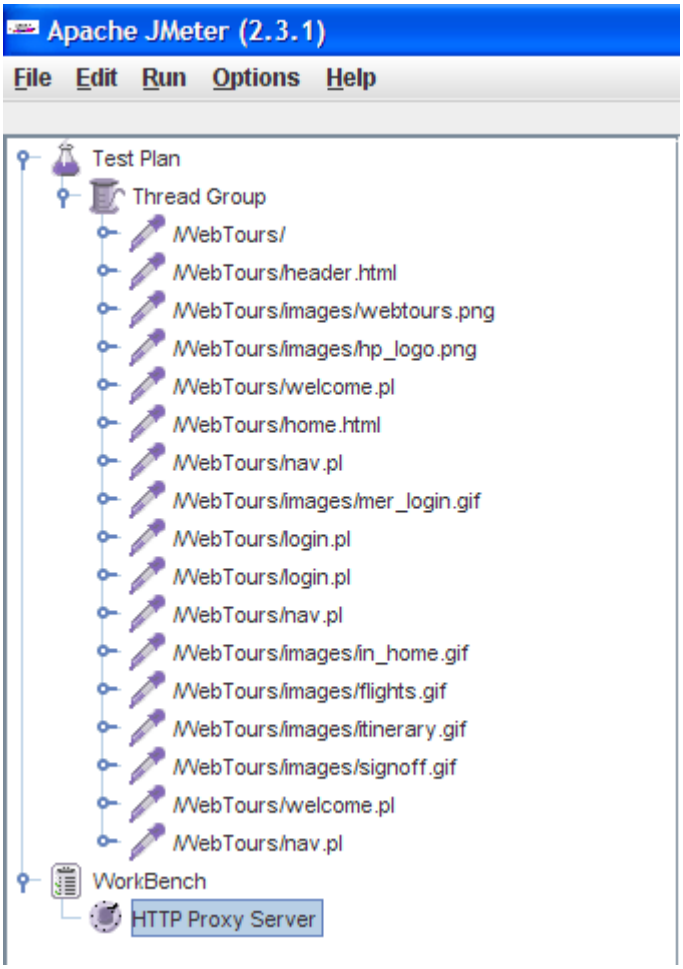
-> “连接(connections)” -> “局域网设置(LAN setting)”，如下图所示：



在局域网设置(LAN setting)界面勾上“为 LAN 使用代理服务器(Use a proxy server for your LAN)”，设置地址(address)为“localhost”，端口(Port)为 8080，确定(OK)。

2.4 录制脚本

在浏览器的 URL 栏输入需要测试的地址，进行操作，操作完毕后，点击 JMeter 中的“Http 代理服务器(HTTP Proxy Server)”的“停止(STOP)”按钮，你将能看到“TestGroup1”下面已经录制了刚才操作的内容。如下图所示：



2.5 样例文件

Record_auto.jmx

3 使用 JMeter 获取 SessionID

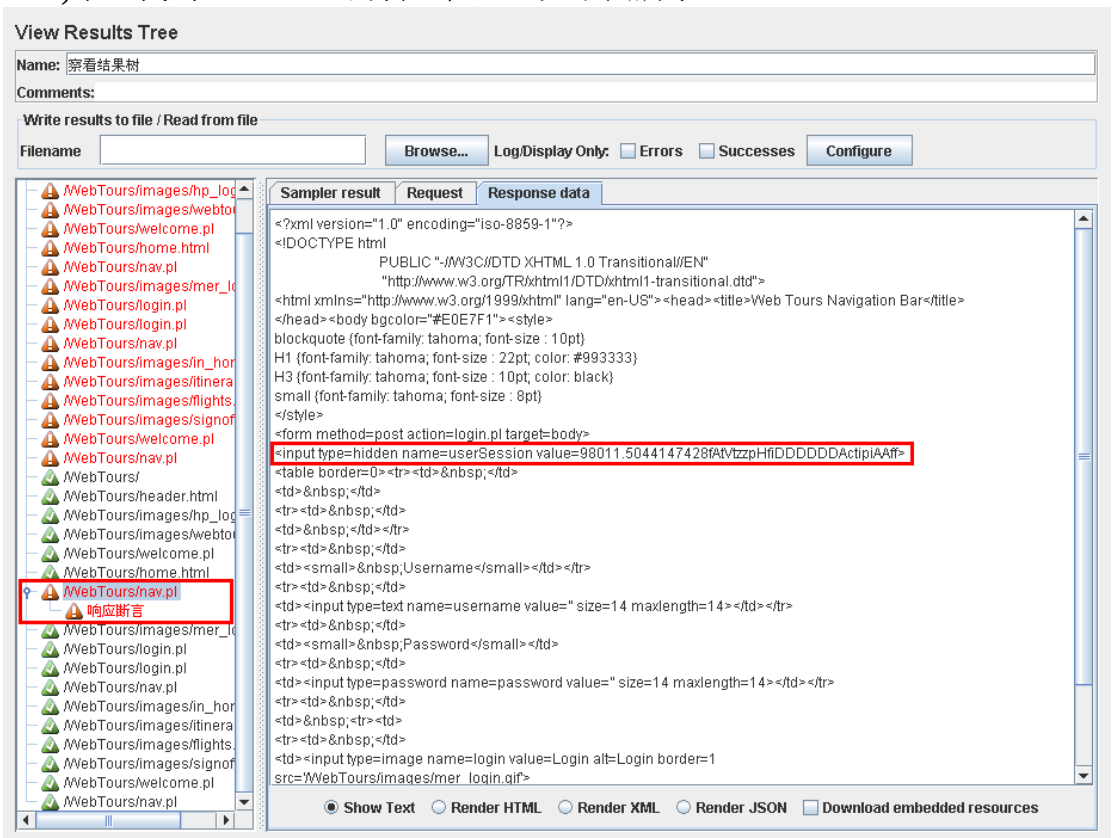
前提：本章所使用的脚本录制请参见《使用 JMeter 自动录制脚本》

3.1 添加察看结果树(View Results Tree)

添加查看结果树，查看脚本运行情况：
Add->Listener->View Results Tree

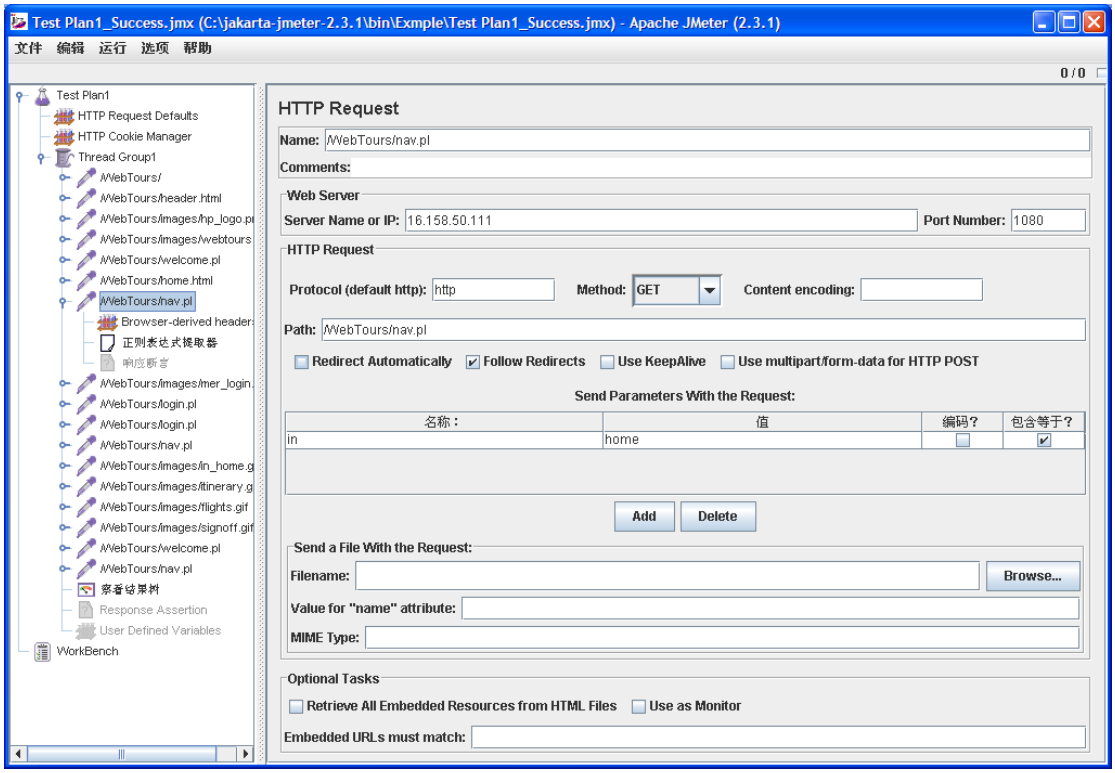
3.2 确认 SessionID 的获得位置

根据录制情况确认 SessionID 的获得位置。我们在查看结果树(View Results Tree)中查找到 SessionID 的传送位置，如下图所示：



根据响应数据(Response Data)的内容可以确定，在本案例中 SessionID 出现在 http://hostname:1080/WebTours/nal.pl 页面中。

PS：由于 nal.pl 页面包含于 http:// hostname:1080/WebTours/index.html 中的子页面，所以需要对页面足够了解或者使用录制工具获得该页面的访问信息（本次测试使用的是 JMeter 的自动录制工具）。这一点区别于 LoadRunner，LoadRunner 可以直接处理包含的页面，不需要单独访问子页面。



3.3 添加正则表达式提取器(Regular Expression Extractor)

正则表达式的提取是在网页下载后进行的，所以正则表达式提取器(Regular Expression Extractor)在后置处理器（POST Processors）中进行选择。

操作步骤（在 nal.pl 页面中）：

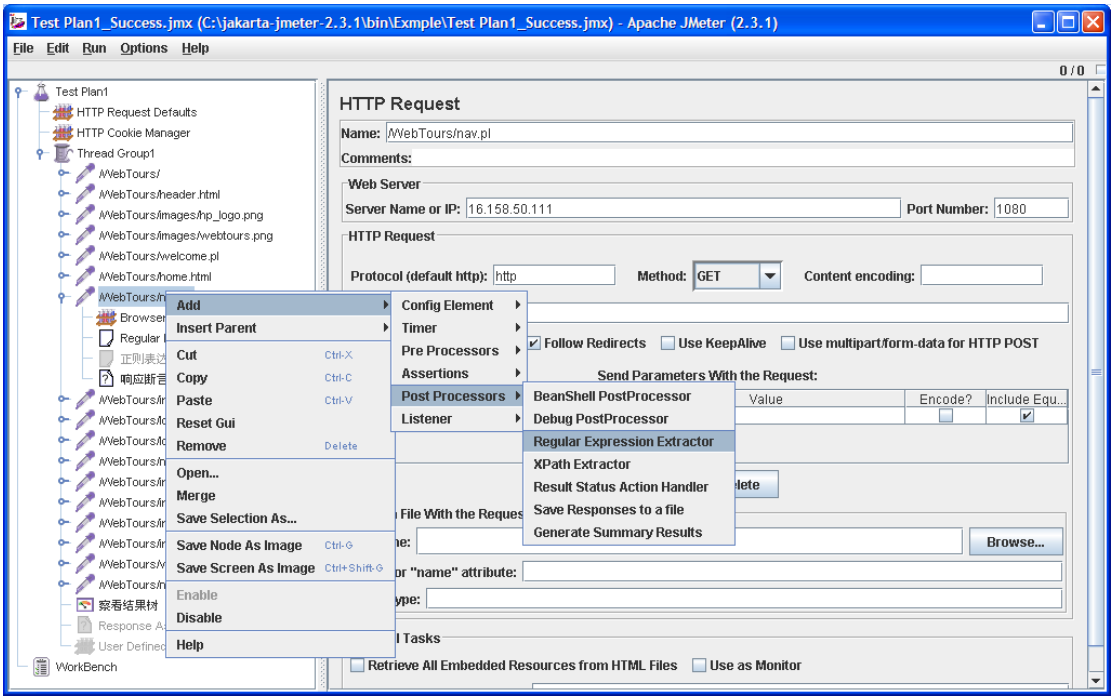
英文版：

右键点击”HTTP Request”中的”/WebTours/nav.pl”->”Add”->”POST Processors”->” Regular Expression Extractor”

中文版：

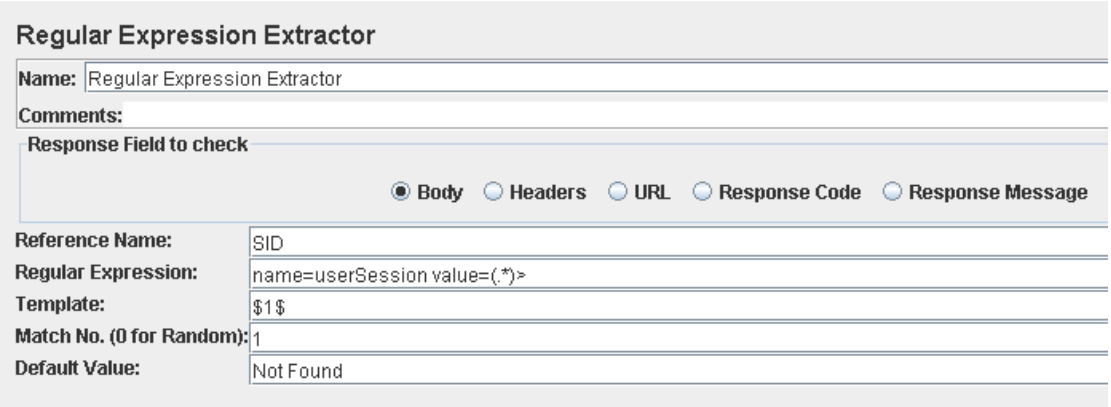
右键点击”HTTP 请求”中的”/WebTours/nav.pl”->”添加”->”后置处理器”->”正则表达式提取器”

如下图所示：



3.4 填写正则表达式（Regular Expression）设置信息

正则表达式（Regular Expression）的设置如下图所示：



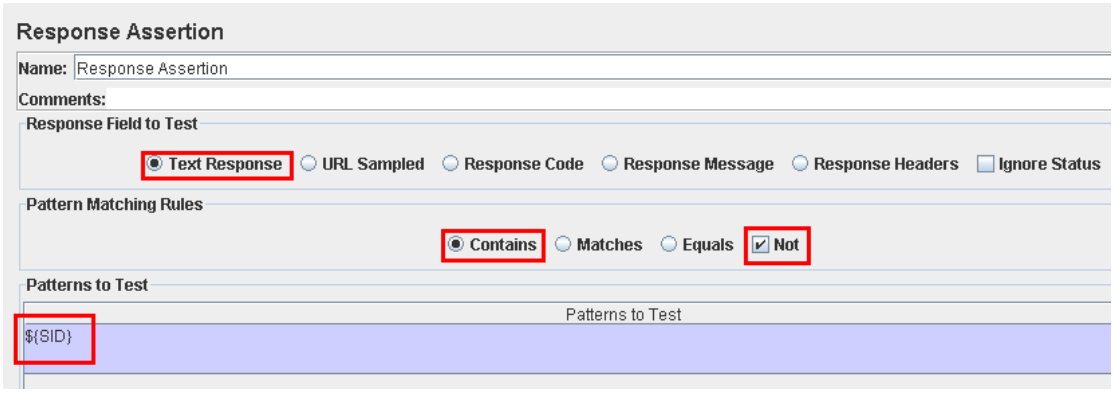
在 Response Field to Check 栏目中选择 Body 项目，在返回网页的主体部分进行查找。

Attribute	描述	值
Reference Name	引用名称，基本上可以理解为变量名	SID
Regular Expression	用于匹配的正则表达式。(.*)为需要匹配的单元	name=userSession value=(.*)>
Template	如果正则表达式匹配多个值则需多多个值进行分组，本次我们仅使用一个组	1
Match No.	确认第几项匹配的是我们要选的值。 等同于 LoadRunnner 脚本 web_reg_save_param 函数的 Ord 属性设置	1
Default Value	在没有取到值的时候使用的默认数据	NotFound

这里只是单个匹配的情况，如果要出现多个匹配建议修改正则表达式，也可以按匹配顺序引用数据，请参见 JMeter User Manual 内容。

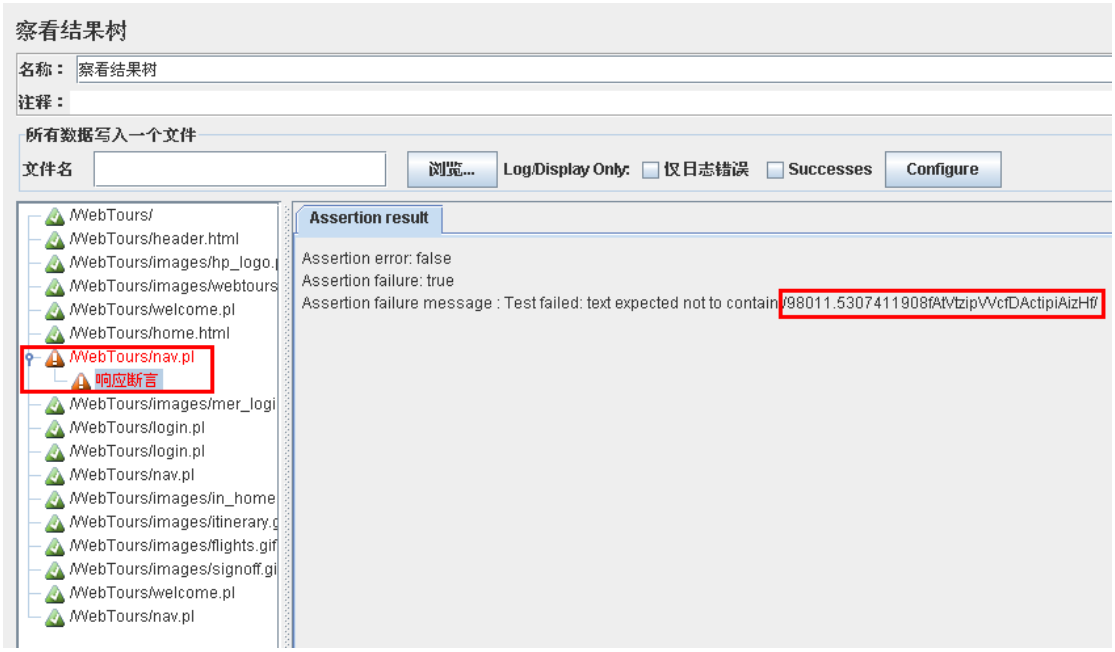
3.5 添加断言(Response Assertion)检查匹配结果

为确定是否正确提取 SessionID 值，在”HTTP Request”(/WebTours/nav.pl)中添加断言，设置如下图所示：



Response Field Test 中选择 Tex Response 对接受的文本信息进行检验。

Pattern Matching Rules 中选择 Contains，检验文本是否包含检验信息。选择 Not，因为如果判断正确在 View Results Tree 看不到结果，所以使用 No，这样在正确的时候将显示错误，测试人员可以在 View Results Tree 中查看结果，如下图所示：



Pattern to Test 输入匹配数据${SID}，此变量对应于前面正则表达式（Regular Expression）所获取的变量名称。

3.6 样例文件

Test Plan1_Success.jmx

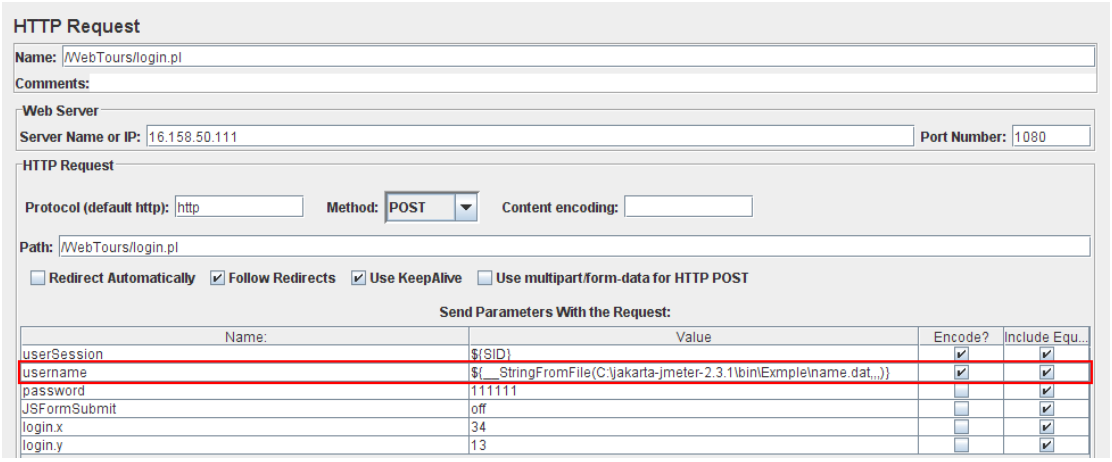
4 JMeter 参数化过程 1

前提：本章所使用的脚本录制请参见《使用 JMeter 自动录制脚本》

4.1 选择函数位置

参数化是指在进行性能测试的过程中使用不同的参数来模拟系统的处理性能，从而使压力测试结果更加接近实际情况。

在本案例中，我们通过 JMeter 模拟多用户同时登录系统。根据对录制脚本的分析，用户登录名通过 POST 模式进行传递，所以测试人员确定位参数传递页面是 HTTP Request(/WebTours/login.pl)中实现。如图所示：



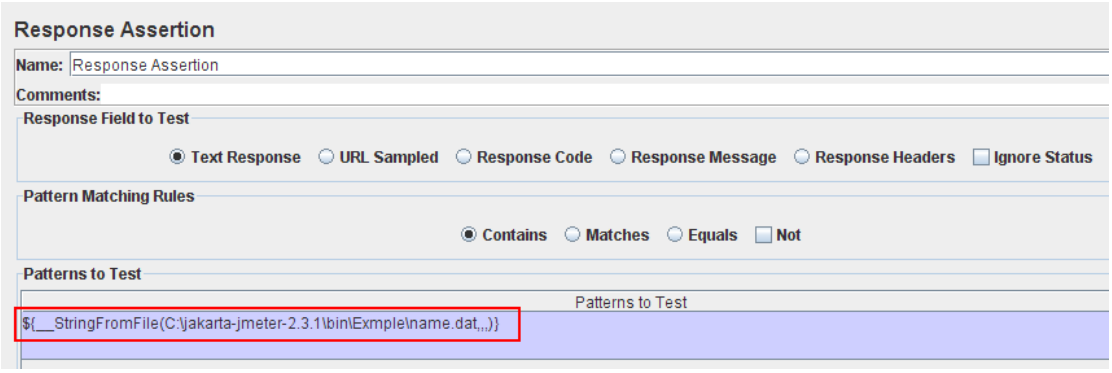
我们将 username 的 value 使用函数进行替换。

4.2 添加_StringFromFile 函数

_StringFromFile，这个函数是从一个文件中取到一个字符串，这个函数和 LoadRunner 中的 File 变量差不多，不过 LoadRunner 可以直接从数据库中查询记录，而 JMeter 需要我们借助第三方工具生成文本文件。生成文本文件后就可以设置这个函数的参数了，参数分别是

Name	Description	Value	必填
Enter full path to file	文件的完整路径，即文件路径＋文件名.扩展名（可以有空格和中文）		√
Name of variable in which to store the result	参数的名字用于和其他同类参数的区分		╳
Start file sequence number	参数是文件开始的序号，也就是文件读取的其起始行数。		╳
Final file sequence number	参数是文件的结束序号，也就是要读取文件的最后行。		╳

_StringFromFile 函数也可以在其他位置直接使用，如断言(Assertion)位置：



4.3 数据文件制作

数据文件使用标准的 TXT 文件格式，每一行保存一个参数供函数调用使用。

name.dat

文件内容如附件所示：　　。

PS：在使用过程中根据实际存储情况更改“Enter full path to file”参数的设置路径。

4.4 补充说明

如果并发人数和循环次数超过了文件内容，那最后一次循环读取的文件内容和第一次的一样，函数会自动循环读取。文件的起始序号和结束序号也可以不用设置，这样函数会从第一行读取到最后一行，然后再循环读取。

4.5 样例文件

Exmple_StringFrom File.jmx

5 JMeter 参数化过程 2

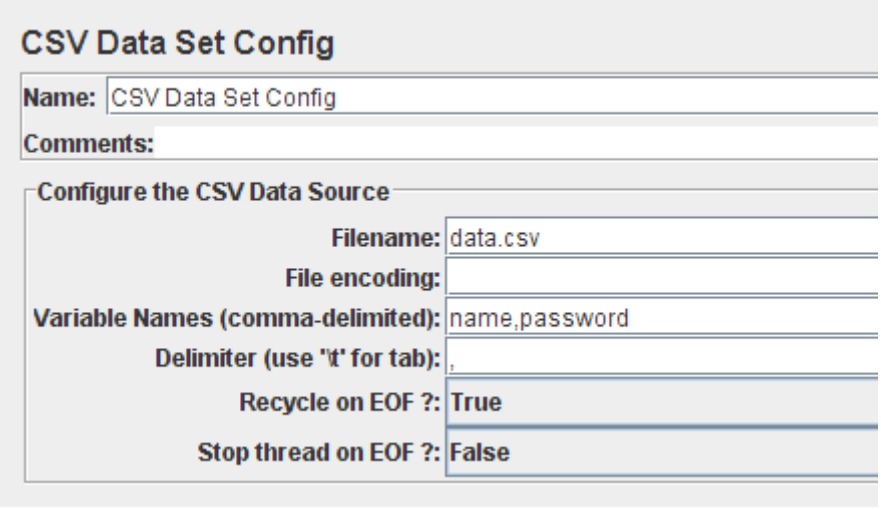
5.1 选择函数位置

参见 4.1 选择函数位置

5.2 添加 CSV Data Set Config

CSV Data Set Config 可以将数据由 CSV 格式文件中读出，并保存为变量，以便测试工程师在脚本过程中调用。

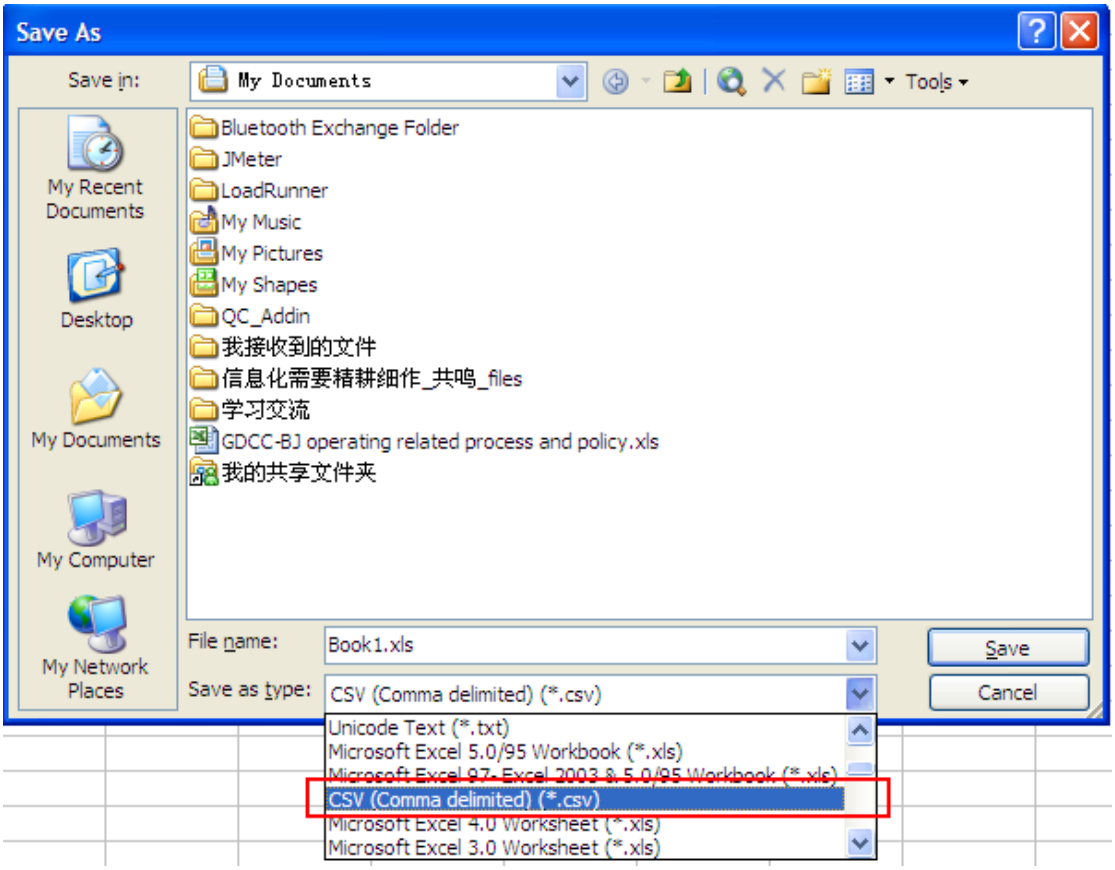
在线程组中插入 CSV Data Set Config，并设置以下参数如图所示：



Name	Description	必填
Filename	数据文件的文件名	√
File encoding	数据文件的编码格式	╳
Variable Names	变量名称，多变量时使用逗号分割不同变量	√
Delimiter	在数据文件中数据的分隔符	√
Recycle on EOF	当数据文件中的数据使用完毕是否循环使用数据	√
Stop thread on EOF	当数据文件中的数据使用完毕是否中止线程	√

5.3 数据文件制作

基本的 CSV 文件可以通过 Excel 保存为 CSV 文件获得，如下图所示：



与 Load Runner 不同的是 JMeter 不判断字段名成，所以如果数据文件带有字段名称请删除字段名称所在的第一行，以保证脚本可以正确读取到参数。

data.csv

本用例使用的文件：

请注意数据文件必须和测试计划文件(*.jmx)保存在同一目录下，JMeter 才可以正确读取数据。

5.4 样例文件

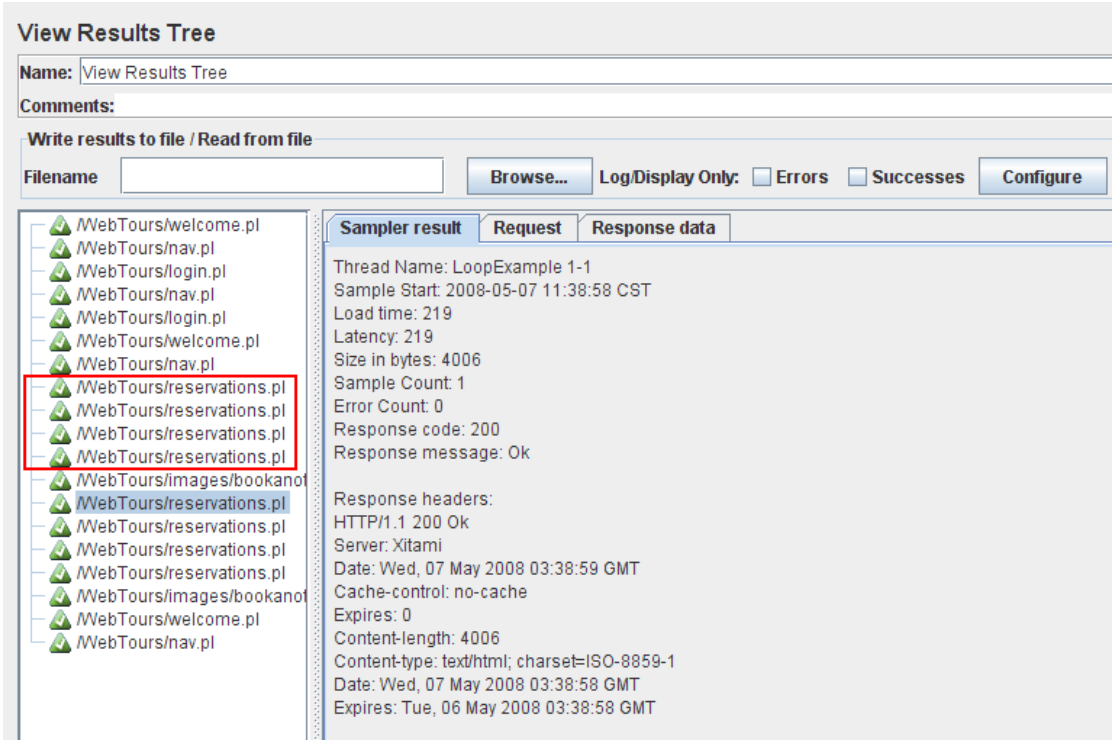
CSV Data Set Config.jmx

本文件中使用的数据文件 tester02 和 tester04 登录密码为错误密码，通过断言(Assertion)可以看到登录失败，验证参数可以正常替换。

6 JMeter 逻辑循环添加

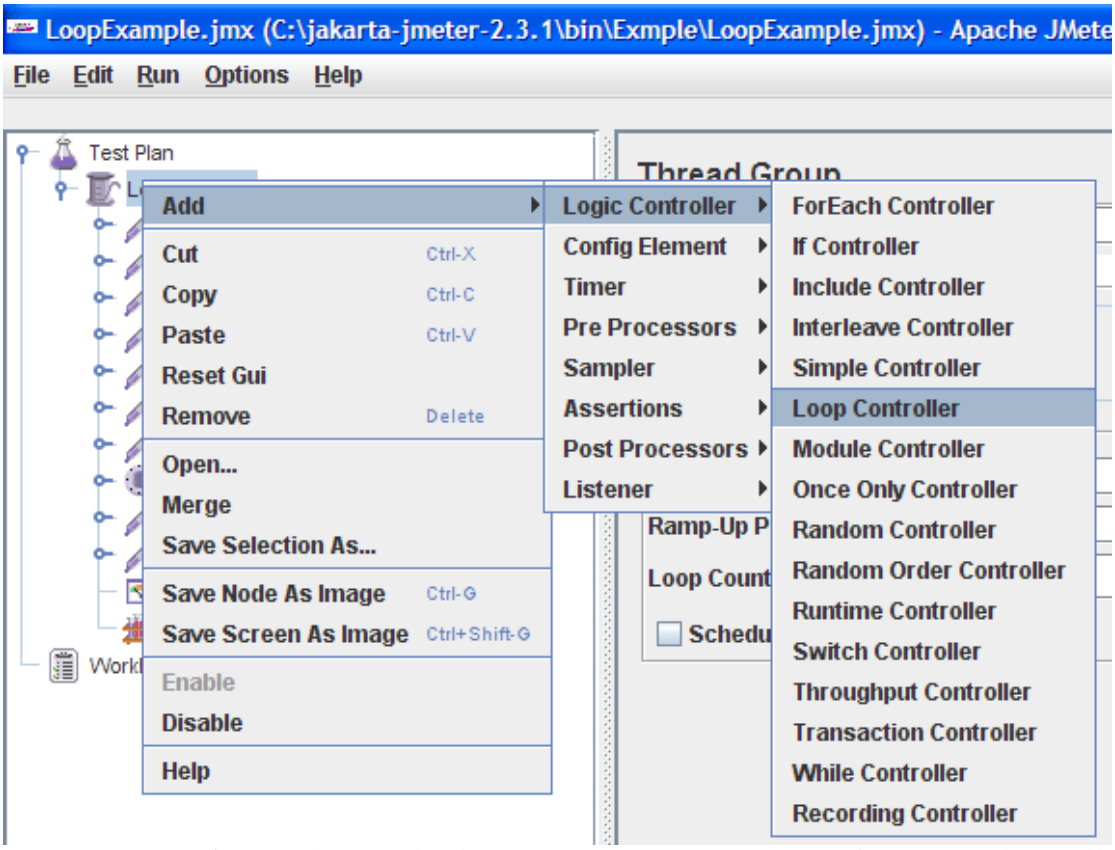
6.1 判断循环内容

通过阅读结果树(View Results Tree)的内容，由测试工程师根据测试需求判断循环业务所在位置。在本用例中，我们将机票的定购过程作为循环内容。如下图所示：



6.2 添加逻辑循环控制器

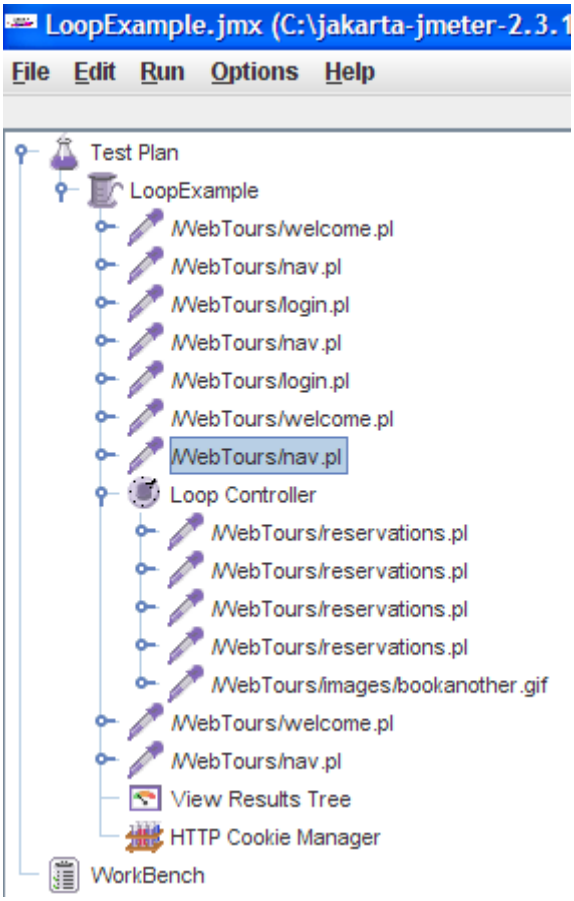
在线程组(Thread Group)中添加循环控制器(Loop Controller)，如下图所示：



我们根据实际业务需要在循环控制器(Loop Controller)中设置的循环次数(Loop Count)中输入数字。

6.3 重新安排 HTTP 请求(Http Request)

在 JMeter 中单元的作用范围以树的形式进行表现，并按由上至下的顺序执行，所以添加完循环控制器(Loop Controller)后要对其位置和作用范围进行相应的调整。如图所示：



循环控制器(Loop Controller)设置在 Http 请求(/WebTours/nav.pl)后，将需要循环的业务内容(/WebTours/reservations.pl)作为子项目拖拽到循环控制器(Loop Controller)中。

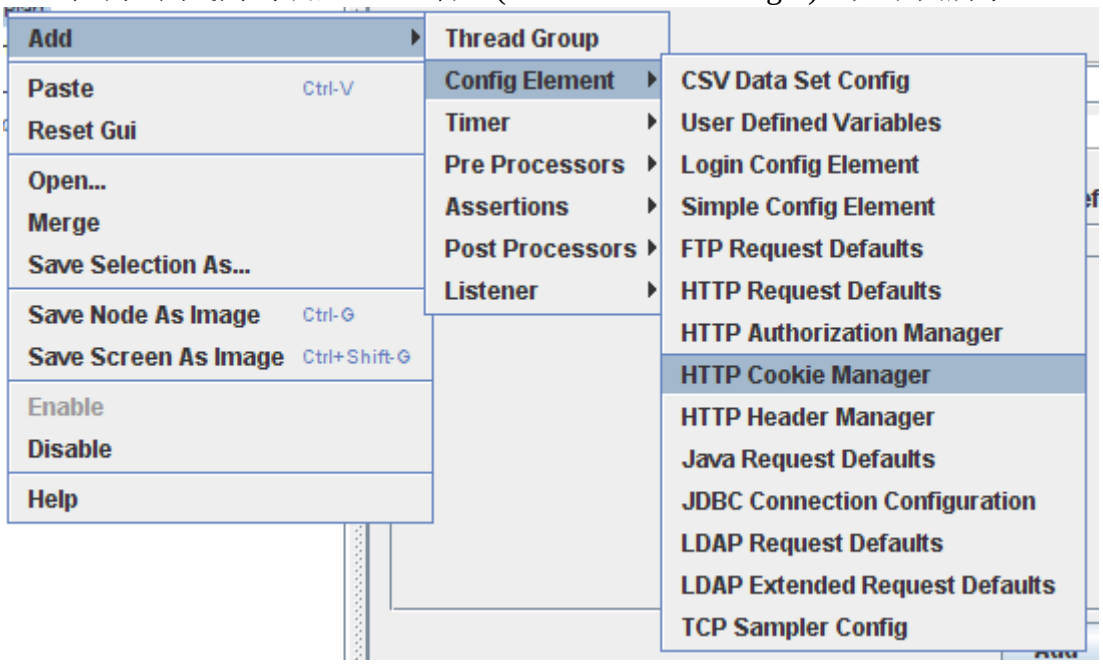
6.4 样例文件

LoopExample.jmx

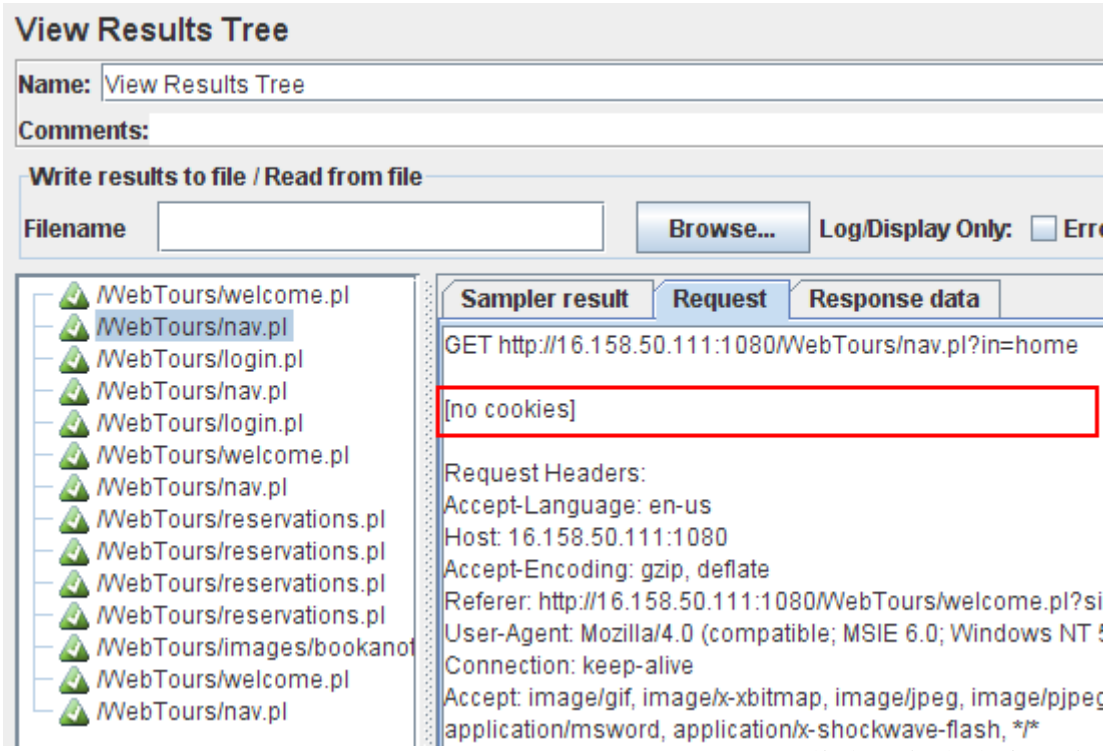
7 HTTP Cookie Manager 的应用

7.1 添加 HTTP Cookie Manager

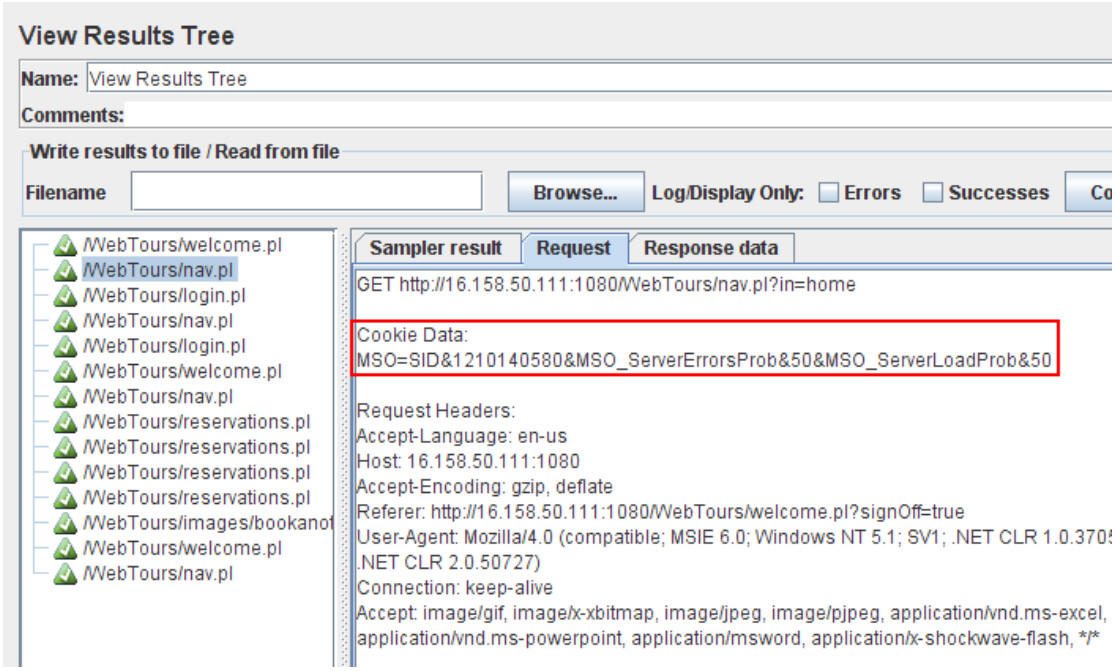
在测试计划中添加 Cookie 管理(HTTP Cookie Manager)，如下图所示：



JMeter 可以存储线程所对应的 Cookie 信息，并在脚本中应用这些信息。未添加 HTTP Cookie Manager 的脚本，JMeter 无法识别 cookie 信息，如下图所示：

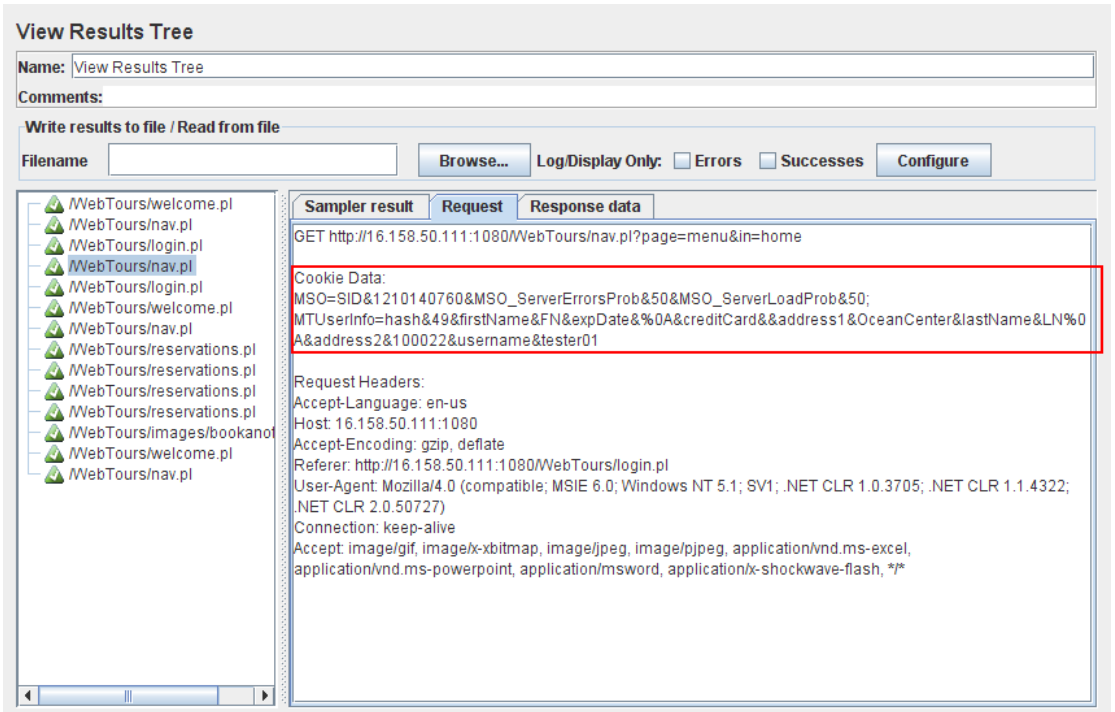


添加 HTTP Cookie Manager 后，JMeter 可以识别 cookie 信息并保存在变量中，以供使用，如下图所示：



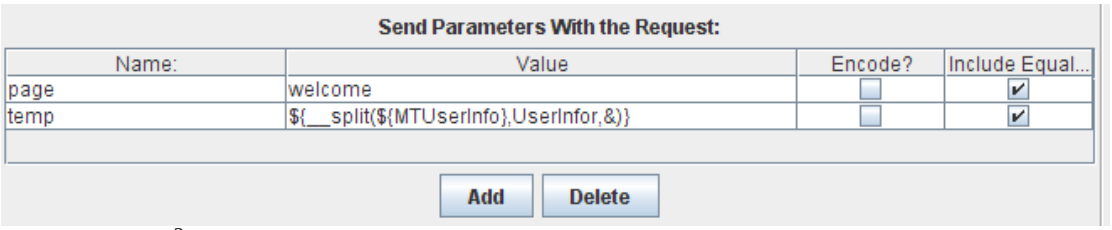
7.2 Cookie 信息的处理

在本用例中，用户的基本信息被存储在一个变量中，如下图所示：

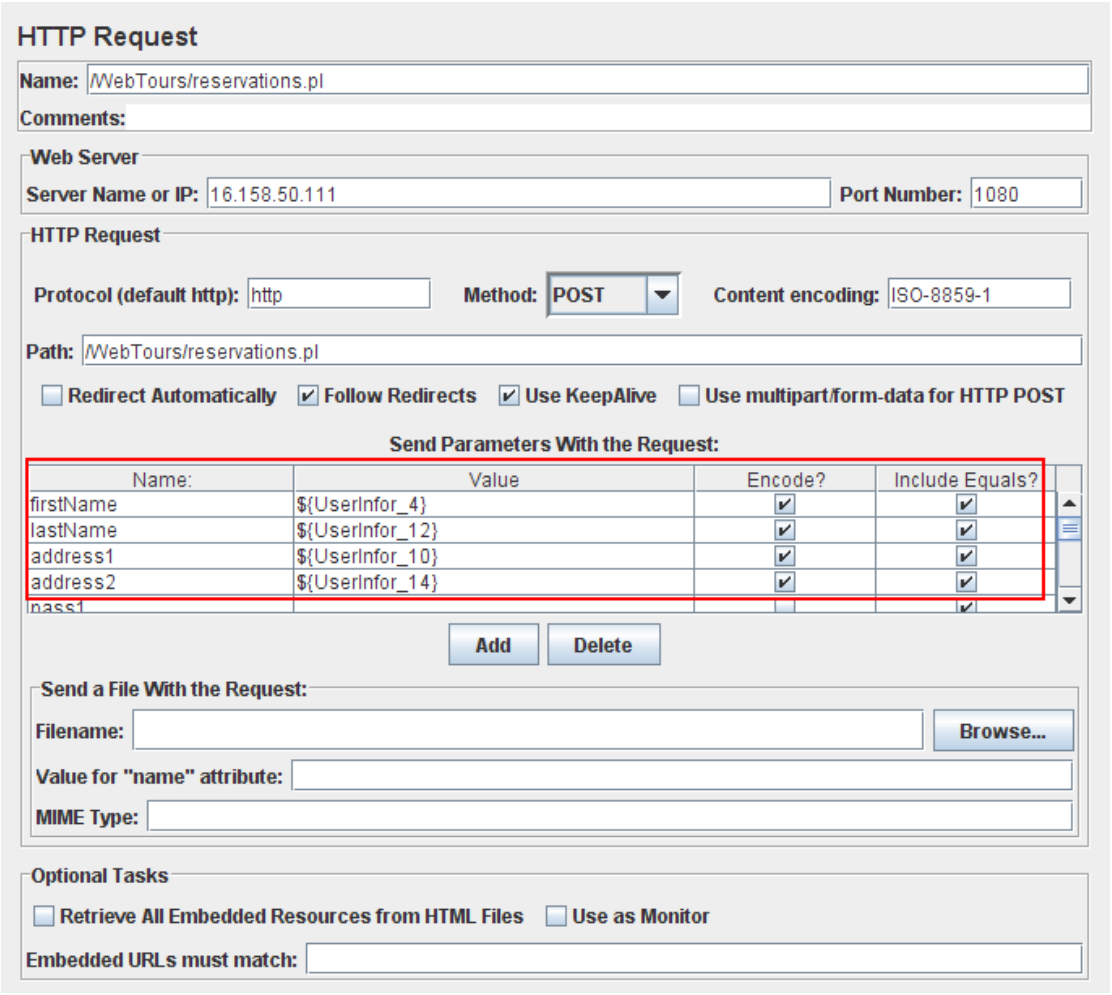


在 Web 应用的过程中我们需要分别引用用户信息，所以引入__split()[1]函数对保存有用户信息的变量 MTUserinfo 进行分解。我们在获得 Cookie 后将其进行分解如下图所示：

[1] __split()的详细使用请参考《JMeter User's Manual》



__split()[2]函数将 MTUserInfo 分解为一系列变量，测试工程师根据自己的需要选用变量使用，如图所示：



7.3 样例文件

CookieExample.jmx

[2] __split()的详细使用请参考《JMeter User's Manual》

8 JMeter 结果处理

8.1 添加察看结果树(View Results Tree)

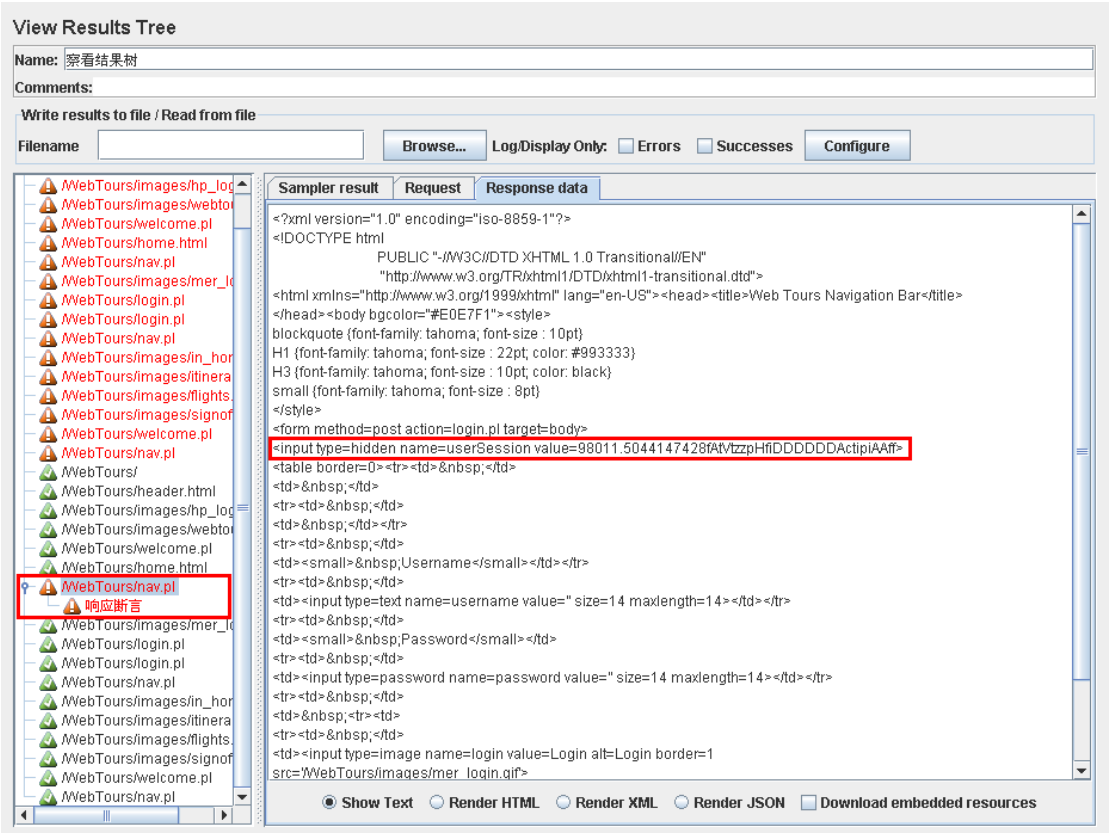
添加察看结果树(View Results Tree)，可以察看脚本运行情况。在测试初期，工程师调试脚本并观察运行脚本的执行效果都是通过察看结果树(View Results Tree)进行的。它主要分为三种试图：

取样器结果(Sampler result)：用于察看 Http 请求(Http Request)的执行情况。

请求(Request)：察看 Http 请求(Http Request)发送情况，可以在这里察看 POST 参数和 Cookie 的内容信息。

响应数据(Response data)：可以查看客户端所得到的响应数据(网页)内容，可以文本模式察看，也可以使用网页等形式察看。

如图所示：

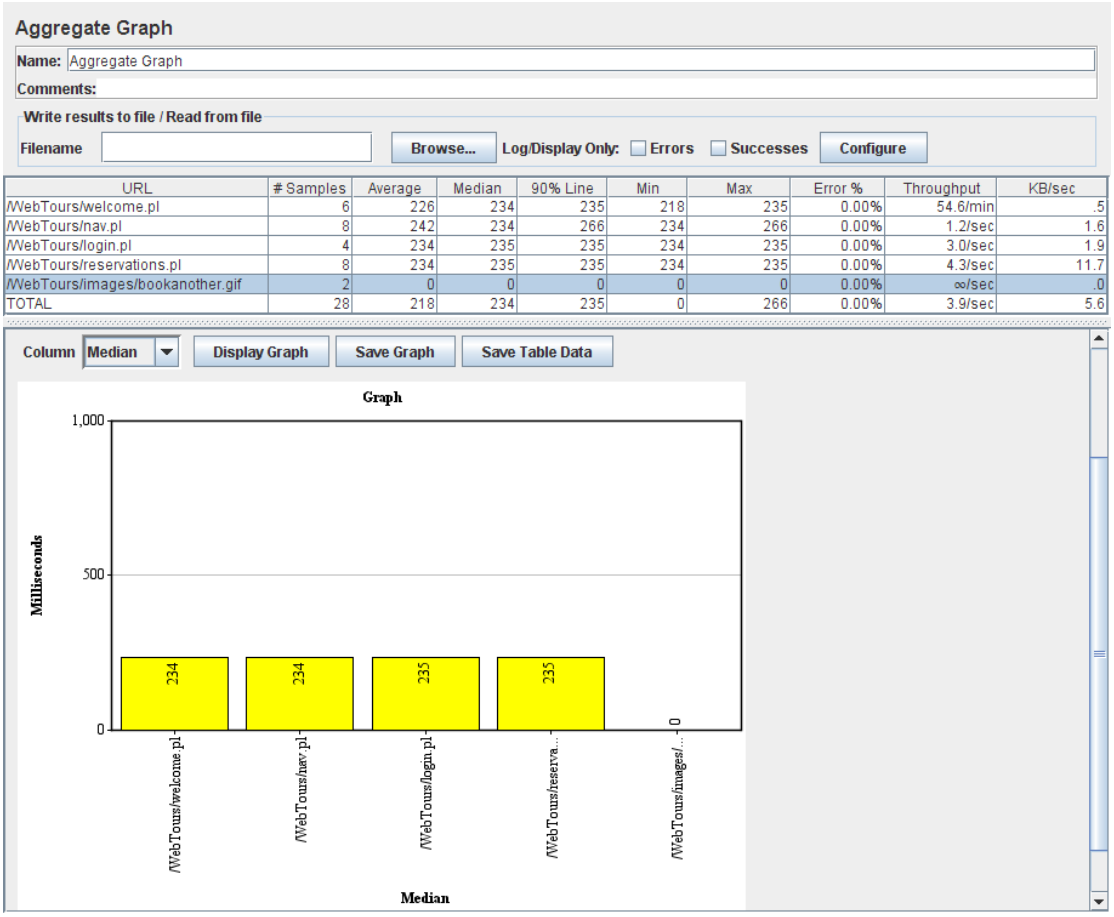


添加操作：

Add->Listener->View Results Tree

8.2 Aggregate Graph

添加 Aggregate Graph 使测试人员可以察看测试计划中所有的取样(Sampler)的响应时间的均值，并可以将数据保存为文本格式和图像格式。



需要注意的有两点：

1. Aggregate Graph 通过每一个取样器(Sampler)的名字进行归类，所以工程师在录制完成脚本后，要根据统计需要重新对各 Sampler 命名以保证数据准确。
2. Aggregate Graph 在每次执行测试计划的时候不能自动清空，如不清空会造成测试结果数据的累加，所以需要测试人员在执行测试计划前手动清空其中的数据。

8.3 样例文件

ListenerExample.jmx