# 1．接口概述

在与研发人员沟通过程中，经常会听到这个值是通过xx接口传递的；这里有个Bug，你看下xx接口调试下；系统要预留xx接口。这接口到底为何物呢，今天就来为大家介绍下接口（包括接口测试），让大家看到接口不再陌生。

## 1.1什么是接口

计算机中包括硬件接口和软件接口。电脑等信息机器硬件组件间的接口叫硬件接口，是可以看到的以实物存在的如串口、并口等；而电脑等信息机器软件组件间的接口叫软件接口。而软件接口则是虚拟存在的接口。

接口广义的定义为：泛指实体把自己提供给外界的一种抽象化物（可以为另一实体），用以由内部操作分离出外部沟通方法，使其能被修改内部而不影响外界其他实体与其交互的方式。

接口狭义的定义为：是指特定的函数集合，一般是用interface（Delphi）声明的，它表示一个方法集合，这个集合被成为一个命名接口。一个命名接口中的方法必须在一个类中实现后才能被使用，一个类继承实现一个接口，称为这个类实现了该接口，一个接口可以被多个类实现，一个类也可以实现继承多个接口，这样就形成了一种灵活的接口调用的方式，从而实现更加灵活和节省资源的多态。

这里说下我个人对接口的理解：接口就是提供一个入口或者提供一个方法来改变要调用的对象的属性，或者得到一些想要的值。

目前测试接触到的接口基本都是以HTTP协议为基础的接口（包括WebService接口）。

## 1.2什么是接口测试

接口测试是项目测试的一部分，正如其名，它测试的主要对象是接口，是测试系统组件间接口的一种测试。接口测试主要用于检测外部系统与系统之间以及内部各个子系统之间的交互点。测试的重点是要检查数据的交换，传递和控制管理过程，以及系统间的相互逻辑依赖关系等。

不管是何种接口测试，其测试都为客户端发送request请求，接着服务器会返回response报文，然后我们需要对response内容进行比对，从而来中判定接口访问是否成功，最终验证业务是否符合需求。

## 1.3为什么要做接口测试

早起的系统业务逻辑相对简单，基本的功能测试、性能测试、GUI自动化测试已经足以覆盖项目的需求。而随着产品功能越来越多，系统架构越来越复杂，一些预想不到的Bug出现在我们面前，怎么办呢？这时候急需要一种更有效的测试方法来适应当前的变化，来持续的保证我们产品的质量，因此接口测试的出现就是为了解决该问题。

从回归测试来说：系统Ａ改了一个接口，相关联系统Ｂ的开发人员并不知道（当然系统Ａ的开发人员也不知道他会影响到Ｂ），导致Ａ发布后，Ｂ出错，Ｂ的用户开始抱怨。此时如果有那么接口测试在持续集成运行的话，当Ｂ测试出错，Ｂ的开发一下就能发现，也就能立即改掉。

另一方面，在测试原则中有这么一条：尽早的和不断的进行测试。阐述的是尽早的持续的测试过程，当进行接口测试时，发现的问题一般都与业务相关，此类Bug造成的危害比UI上发现的问题更为严重，因此有利于Bug的早起修复。

## 1.4接口测试分类

接口测试分为模块接口测试和Web接口测试。模块接口测试需要对代码有一定的掌握能力，可以划分到白盒测试中；而Web接口测试分为服务器接口测试和外部接口测试。

服务器接口测试：是测试浏览器与服务器的接口。这个很容易理解，我们知道web开发一般分前端和后端，前端开发人员用HTML/CSS/JavaScript等技术。后端开发人用PHP/JAVA/Python/Ruby等各种语言。用户输入的数据是输入到的前端页面上，怎样把这些数据传递的后台的呢？通过HTTP协议的GET与POST请求来实现前后端的数据传递。这也可认为是接口测试，调用的登录接口还是查询接口，传参的是用户密码还是搜索关键字。

外部接口测试：这个很典型的例子就是第三方登录，比如你做的新系统免于新用户重新注册的麻烦会提供第三方登录，那用户在登录的时候调用的就是第三方登录的接口，由第三方验证用户名和密码并且返回给当前系统。

## 1.5接口测试目的及方式

核心：保证系统接口的功能正常

方式：持续集成

目的：提高测试效率，保证数据的准确性

文档：接口测试对接口定义文档要求很高，所有的接口数据类型及业务分支导致的报文返回结构是需要事先定义好的，所以要形成文档的习惯，以方便查阅，尽量减少团队与团队间的沟通成本。

同样我们在接口测试中，也需要根据文档，整理出我们的接口测试数据及接口测试案例，有效的生成相关测试报告，方便其它人去审核、分析接口测试的成果。

# 2．接口测试用例设计

## 2.1测试用例设计流程

首先，明确出发点。和所有的测试一样，接口测试出发点是你要证明所测的程序是错误的。以这个出发点为导向，你的设计行为就会尽量朝这个方向发展，更易发现问题，不会出现大方向的偏差。

其次，选择好测试对象。对于一个系统做接口测试选择好的测试对象是接口测试关键。一个系统有无数的接口，每个接口如果分别测试，那将是很痛苦的一件事情，不光繁琐浪费，而且任何一个内部接口的变动，都将导致我们用例的不可用。这里推荐把整个系统作为一个整体，选择整个系统提供给外部使用、交互的最外层接口作为你的测试对象，以此为测试对象的用例将有很好的健壮性，并且更高效。另外，根据数据的流向，又可将这些最外层的接口分为两类：一类是数据进入系统的接口；一类是数据流出系统的接口。进入系统的接口实际是我们用例的执行调用的接口。可通过变化参数对这些接口进行调用，模拟外部的使用；而流出的接口则是我们用例真正该验证的点。数据从哪里流出，流出时的状态如何，此时系统又是什么状态都是我们所应该验证的。

然后，确认完整的测试对象的功能：确认外部接口提供给使用这些接口的外部用户什么样的功能，外部用户真正需要什么样的功能。此两个功能一定要准确详细，用例的设计要严格按照测试对象功能设计才是正确的用例。

最后当出发点、对象、功能都确定了，就可以真正设计用例了。

## 2.2测试用例设计原则

通常，设计接口测试用例需要考虑以下几个方面：

* 是否满足前提条件。有些接口需要满足前置条件，才可以成功获取数据。
* 是否携带默认参数值。带默认值的参数都不填写，必填参数都填写正确，其他不填写。
* 业务规则、功能需求。根据实际的业务规则和需求进行用例设计。
* 参数是否必填。异常情况，参数为空的测试。
* 参数之间是否存在关联。若参数之间存在约束关系，则要设计相关用例进行测试。
* 参数数据类型限制。参数值限制了数据类型，则输入非该类型进行测试。
* 参数数据类型自身的数据范围值限制。根据边界范围设计边界值进行测试。

## 2.3测试用例设计举例

获取订单列表接口。获取店铺指定期间的订单列表---多种条件组合，默认根据日期进行倒序排列。

**消息请求:**字段列表如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **默认值** | **必填项** | **备注** |
| shopid | Int |  | 是 | 商铺编号 |
| token | String |  | 条件 | 设备令牌,Token鉴权方式必填 |
| dateType | Int | 1 | 否 | 订单查询时间字段  1：下单时间：order\_time  2：订单完成时间：order\_finish\_time  3：结算时间：shop\_settle\_time |
| startDate | Date |  | 是 | 查询日期 |
| endDate | Date |  | 否 | 查询结束日期 |
| orderStatus | String |  | 否 | 订单状态，不填表示所有状态  多个状态之间用英文逗号分割  0：已预订；1：已开单；2：派送中；3：已完成(原已结账)；4：退单中；5：已退单；8：自助下单；9：待确认 |
| orderTransactionType | Int |  | 否 | 订单交易状态，不填表示所有  1：未完成；2：已完成 |
| payType | Int |  | 否 | 支付方式，不填表示所有  1：现金；2：POS；3：线上 |
| cashierId | Int |  | 否 | 收银员 |
| billerId | int |  | 否 | 导购员 |
| pNo | Int | 1 | 否 | 页码，从第一页开始，默认为1 |
| pSize | Int | 10 | 否 | 每页记录数，默认为10 |

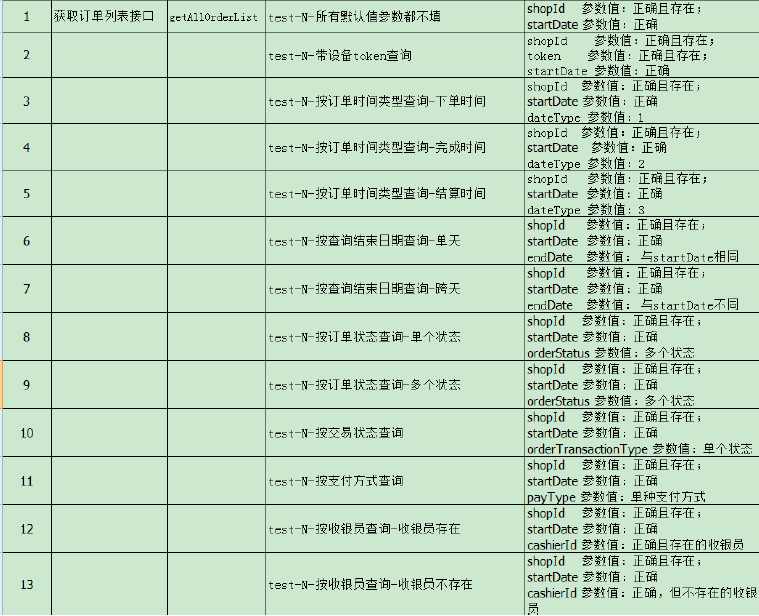
消息请求样例：

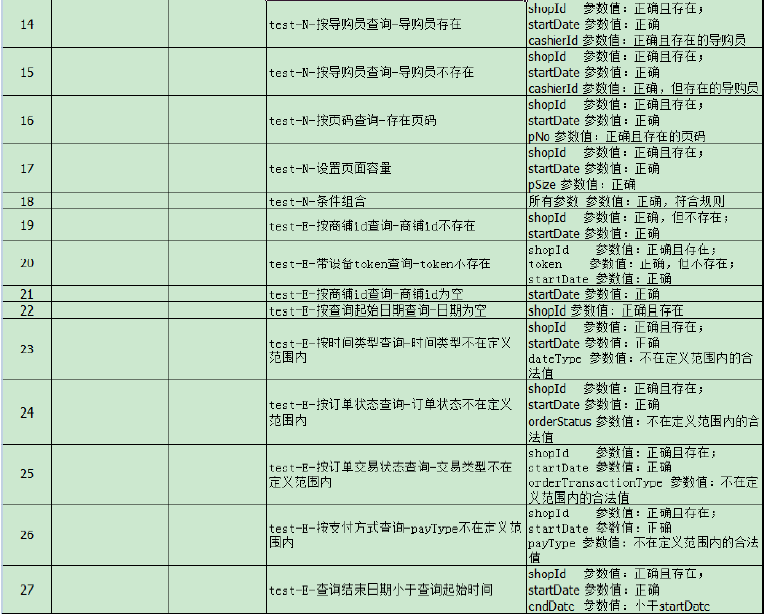
*?shopId=222222&token=123456&queryDate=2015-10-10*

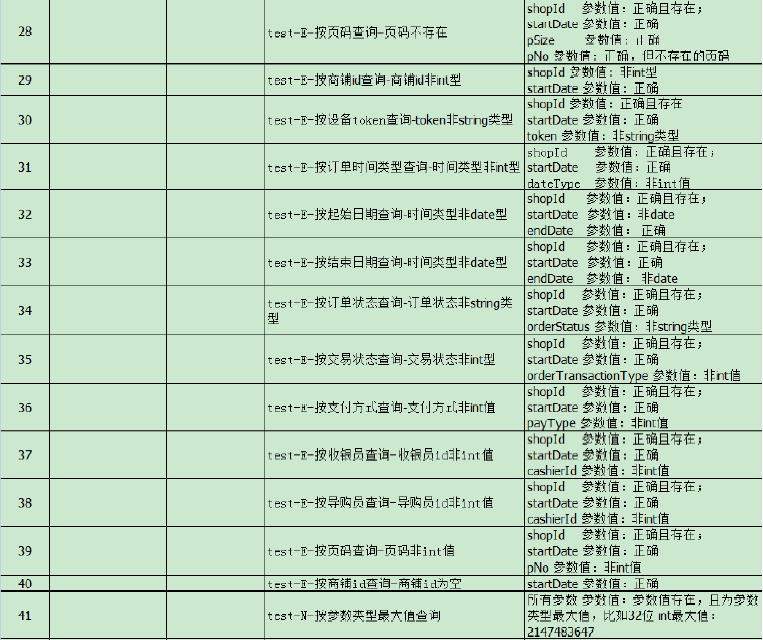
**消息响应：**字段元素如下：

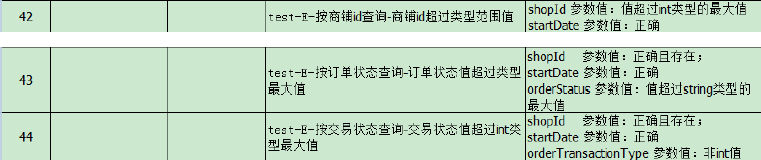
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **默认值** | **必填项** | **备注** |
| orderTotalPriceTotal | double |  | 是 | 实际金额合计 |
| platformTotalIncomePriceTotal | double |  | 是 | 平台服务费合计 |
| cashPayTotal | double |  | 否 | 现金支付 |
| posPayTotal | double |  | 否 | pos支付 |
| onLinePayTotal | double |  | 否 | 线上支付 |
| lst | object |  | 是 | 明细列表 |

部分测试用例如下：









以上测试用例可根据实际情况进行精简

# 3．接口测试工具

## 3.1接口测试工具介绍

接口测试常用的工具包括发包工具和抓包工具。

**发包工具：**

商业工具：LoadRunner、Soapui；

开源工具：Jmeter、Httpclient、python中的urllib2,urllib库。

**抓包工具：**

HTTP抓包工具：HTTP Analyzer 、HTTPwatch、Fiddler、Firebug；

通用抓包工具：Wireshark。

商业工具特点：良好的图形操作界面，良好的技术支持，良好的脚本驱动模式，良好的结果报告，对测试人员的代码能力要求稍低等等，但其缺点也很明显：贵、工具不开源、无法了解问题的本质。

开源工具的特点：大量的资料可以查阅（因为用的人多，社区有大量的人一起贡献），有源码可以查阅，可以根据自已的业务特点进行定制化。缺点就是对测试人员的代码能力有一定的要求，框架需要从零开始搭建。

## 3.2 Jmeter介绍

Jmeter是一款优秀的开源的性能测试工具，然而它不单单可以用在项目的性能测试，同时可以用它来对Web进行接口测试。下面介绍下在接口测试过程中用到的几个模块

### 3.2.1 测试计划

首先打开Jmeter，在菜单左侧出现了“测试计划”。在这里测试计划我们可以把它理解成新建的空白项目，在这个空白项目下面可以添加一系列的接口。

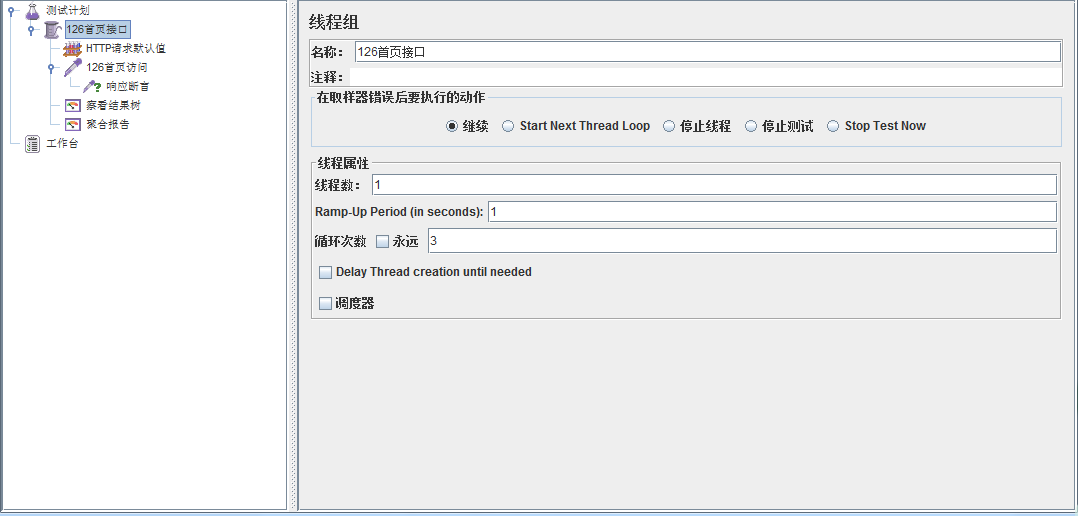


### 3.2.2 线程组

添加方法：测试计划--右键--Threads--线程组。

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

在接口测试中我们可以把它看成一个接口或者一个业务流程组合而成的多个接口集、或者若干个不相干的接口。在线程组中可以设置线程数和循环次数等。

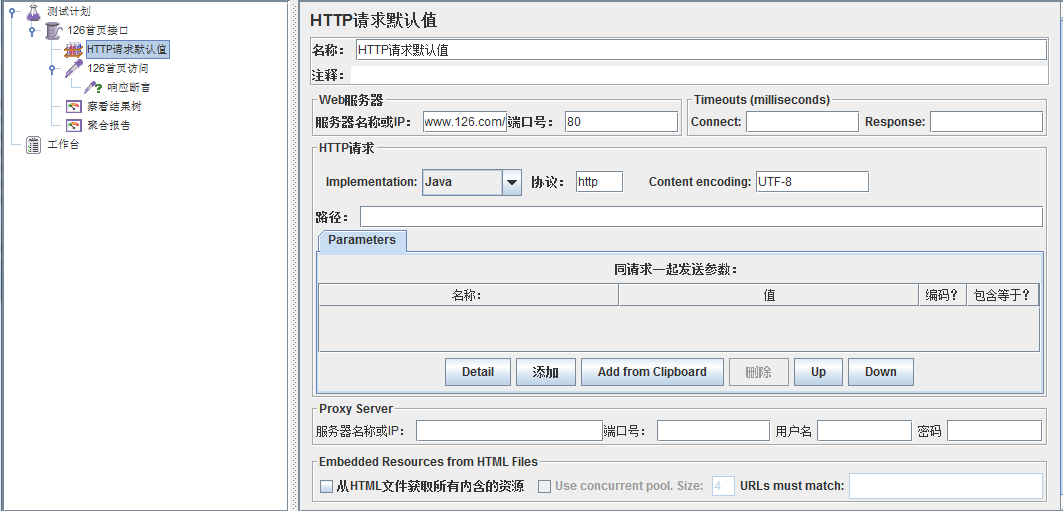


### 3.2.3 HTTP请求默认值

添加方法：线程组 -->右键添加 -->配置元件 --> HTTP请求默认值。

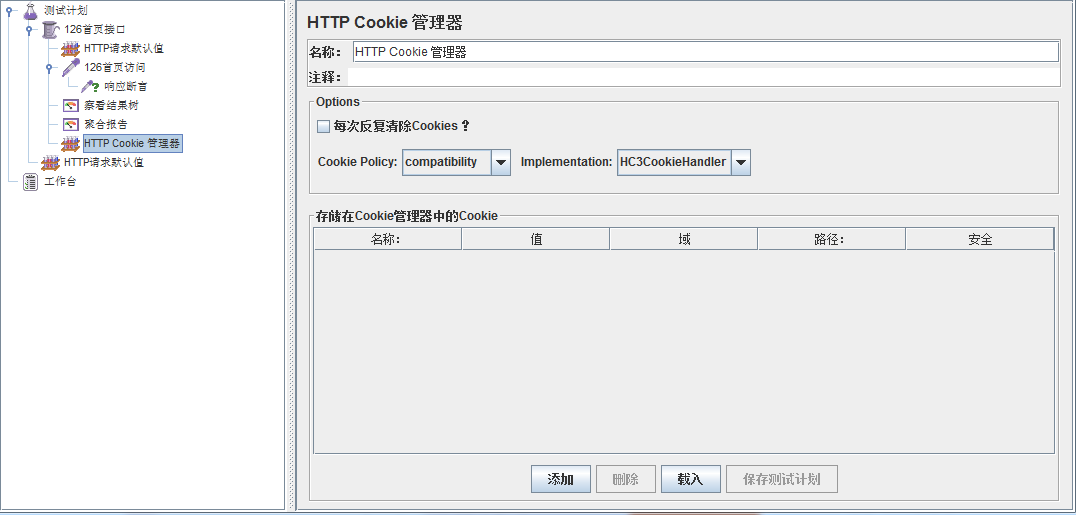
HTTP请求默认值是为了方便填写后续内容而设置。此处填写域名后，后续的HTTP请求中就不用填写地址前缀了。

HTTP请求默认值元件并不告诉JMeter来发送HTTP请求，它仅仅定义这个HTTP请求所用的默认值。



### 3.2.4 HTTP Cookie 管理器

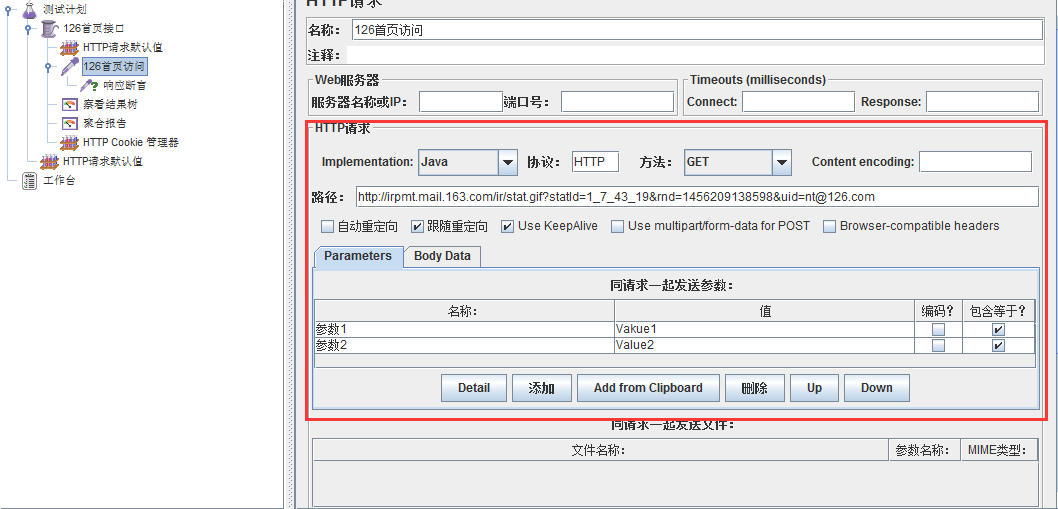
可以像浏览器一样存储和发送cookie，如果你要发送一个带cookie的http请求，cookie manager会自动存储该请求的cookies，并且后面如果发送同源站点的http请求时，都可以用这个cookies。



### 3.2.5 HTTP请求

添加方法：线程组 -->右键添加 -->Sampler -->HTTP请求。

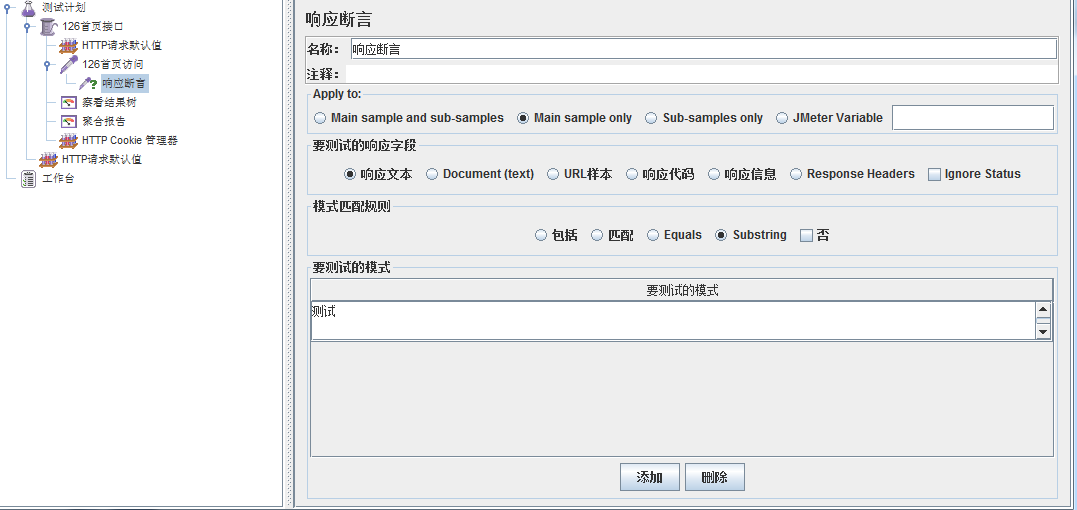
核心内容为HTTP请求部分，包括接口路径和参数，要注意的是如果参数里面用到了其他接口的返回值，这时候需要在Body Data中进行参数化。



### 3.2.6 响应断言

添加方法：HTTP请求 -->右键添加 -->断言 -->响应断言。

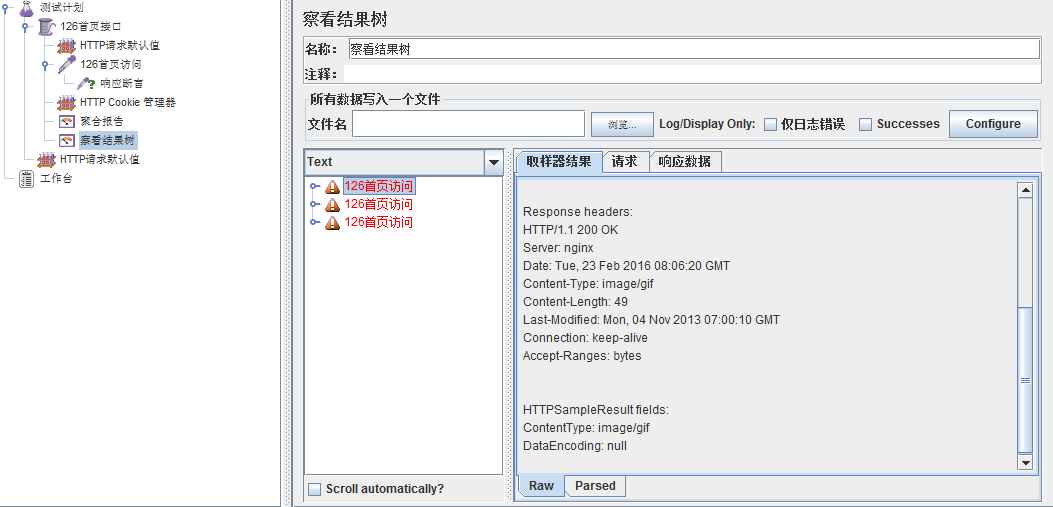
响应断言类似LoadRunner中的检查点，检查接口是否访问成功。如果检查失败的话会提示找不到断言的内容，没提示的话就代表成功了。



### 3.2.7 查看结果树

添加方法：线程组 -->右键添加 -->监听器 -->查看结果树。

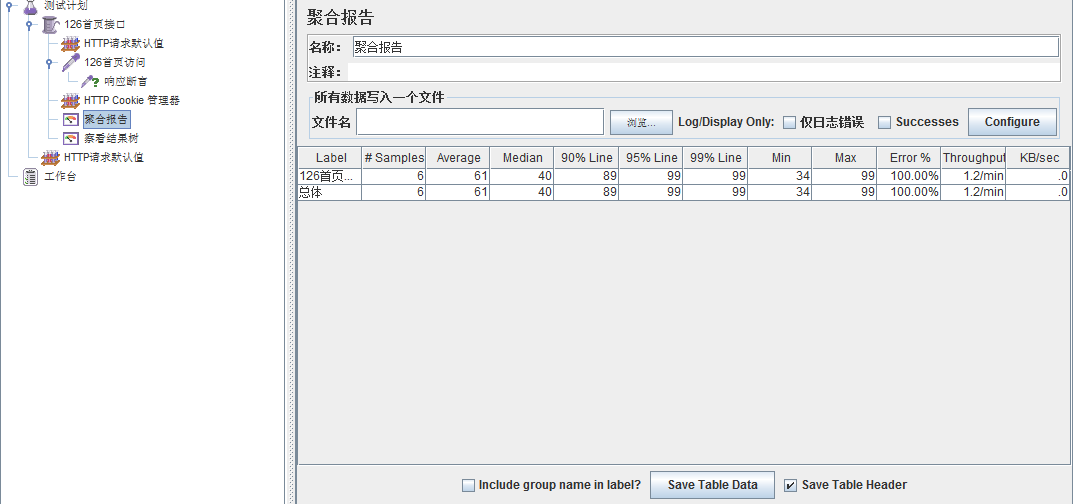
树状形式显示接口的访问结果，包括请求结果、请求内容、服务器的响应内容。



### 3.2.8 聚合报告

添加方法：线程组 -->右键添加 -->监听器 -->聚合报告。

从时间、吞吐量、错误比例等维度统计接口的响应结果信息。



### 3.2.9 数据参数化

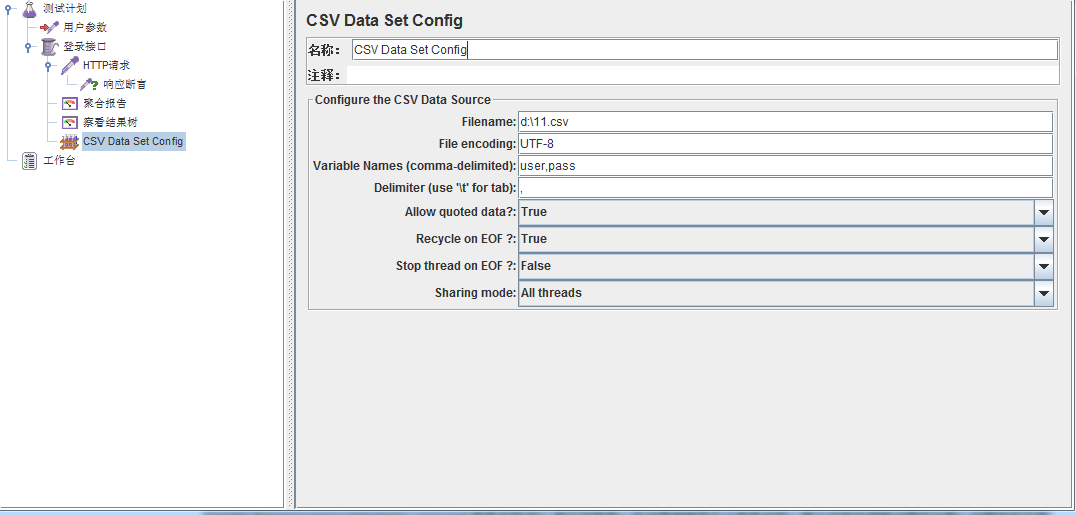
Jmeter中同样可以实现LoadRunner中的参数化功能，有多种方法可以进行操作。下面就来介绍下最简单的。

**①**在需要添加参数的父节点 -->右键添加 -->前置处理器 -->用户参数，填写变量名和变量值，在HTTP请求填写参数值的地方，分别填写：${user}、${pass}即可。如下图：



**②**此种方法针对参数值较少的情况下，若参数取值比较多的话，就需要用使用Jmeter自带函数获取参数值。

采用此种方法之前，需要先添加“CSV Data Set Config”。选中要添加节点(一般是线程组或Sampler元件，区别是作用范围的大小不一致) -->右键添加 -->配置元件 --> CSV Data Set Config，填写好了的CSV Data Set Config截图如下：



①Filename：E:\biz.dat 文件名，指保存参数化数据的文件目录，可以相对或者绝对路径；

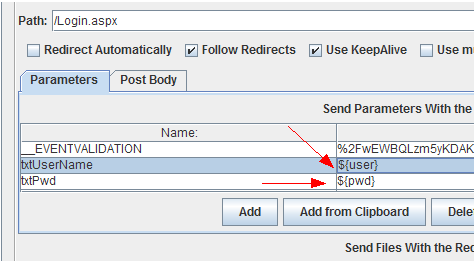
②File encoding:UTF-8，biz.dat文件的编码格式，可以使用电脑自带的笔记本另存为时选择编码方式即可；

③Variable Names(comma-delimited)：user,pwd 参数名称(如：有几列参数，在这里面就写几个参数名称，每个名称中间用分隔符分割，分隔符在下面的“Delimitet”中定义，为了和文件中的“,”对应，这里也用“,”分割每个参数名，这里的 user,pwd，可以被可以利用变量名来引用：${user},${pwd}；

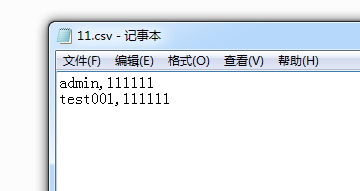
④Delimitet：定义分隔符，这里定义某个分隔符，则在“Variable Names”用这里定义的分隔符分割参数。Allow quote data：选项选为“true”的时候对全角字符的处理出现乱码 ；

⑤Recycle on EOF：是否循环读入，因为CSV Data Set Config一次读入一行，分割后存入若干变量中交给一个线程，如果线程数超过文本的记录行数，那么可以选择从头再次读入。

在HTTP请求的参数值中，对应参数填写对应的参数名：${user},${pass}即可



备注：csv文件的数据格式：1、文件后缀为.csv；2、每一个参数占一列，列与列直接用英文逗号（通常情况下是用英文逗号作分割符，其他符合应该也行，暂没试过。）；3、第一行就开始写参数值，不需要些参数名，如下图：

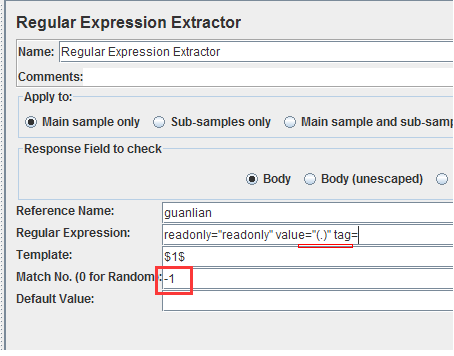


### 3.2.10 数据关联

在LoadRunner中有web\_reg\_save\_param函数可以将服务器的返回值保存到参数中，作为下次访问请求的内容使用。那Jmeter中是否也有该功能呢？答案是肯定有的，可以通过：在需要获得数据的请求上 -->右键添加 -->后置处理器 -->正则表达式提取器。

方法如下：①引用名称即下一个请求要引用的参数名称，如填写title，则可用${title}引用它；②正则表达式中（）括起来的部分就是要提取的。“.”代表任意字符，“\*”代表出现任意次；“+”一次或多次，“?”不要太贪婪，在找到第一个匹配项后停止。③模板，用$$引用起来，如果在正则表达式中有多个正则表达式（多个括号括起来的东东），则可以是$2$，$3$等等，表示解析到的第几个值给title；④匹配数字，0代表随机，-1代表所有，其余正整数代表将在检查的内容中，第几个匹配的内容提取出来。

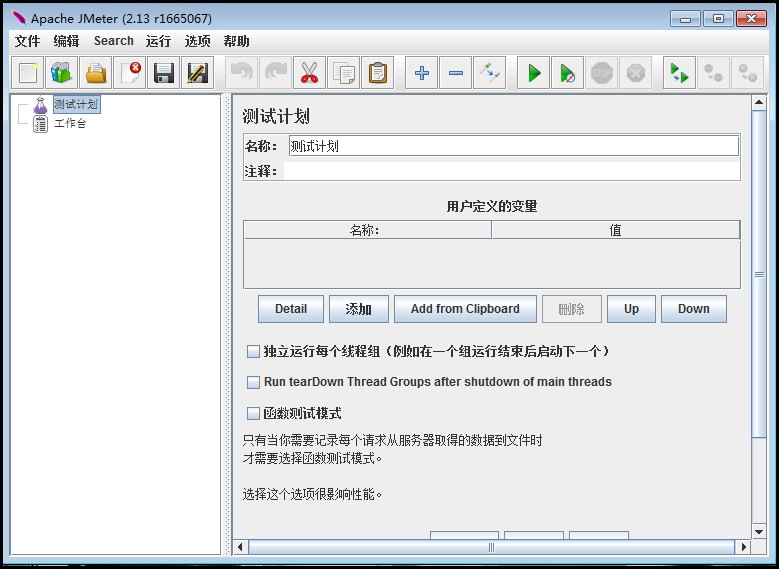
如：<input id="am1" class="am" type="text" onfocus="this.blur();"**readonly="readonly" value="城西分店" tag=**"56964" onclick="hidediv('select1')" />。这是要匹配的串。



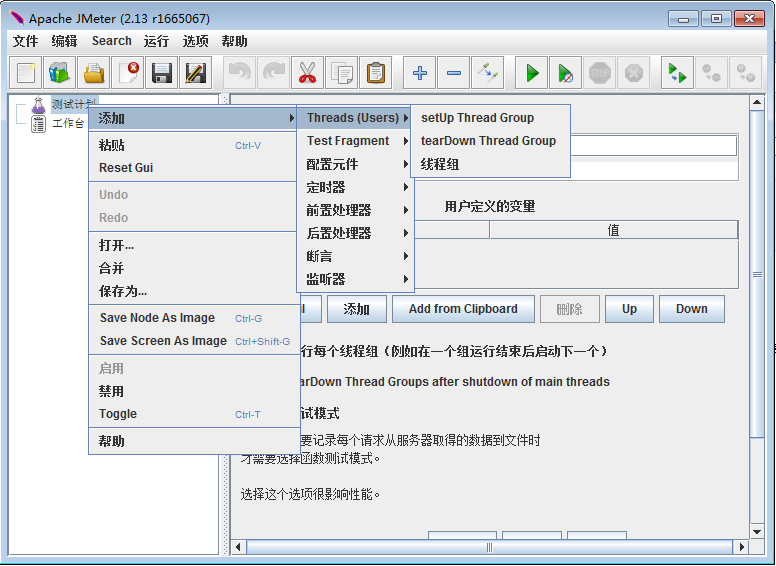
图种Match No.表示匹配第几个找到的值，-1表示所有，0表示随机，1表示第一个，2表示第二个···以此类推

## 3.3Jmeter实战

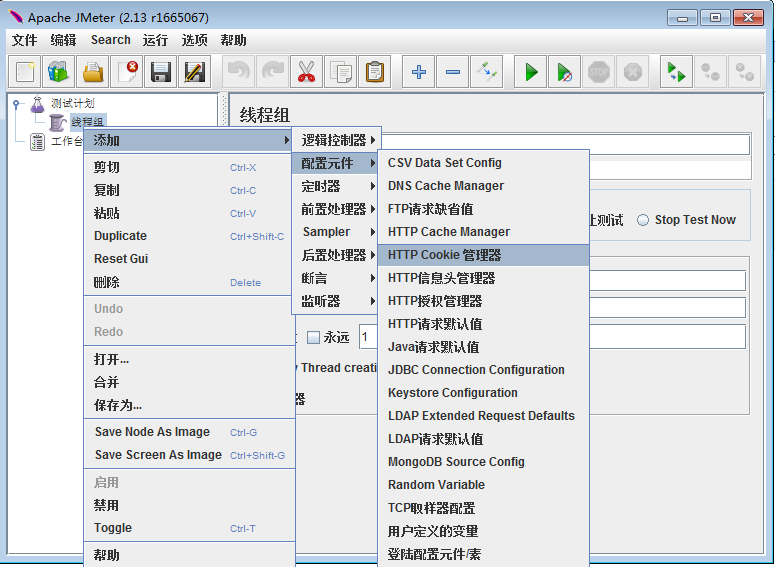
1、打开Jmeter：下载好Jmeter后，双击bin目录下的jmeter.bat文件：



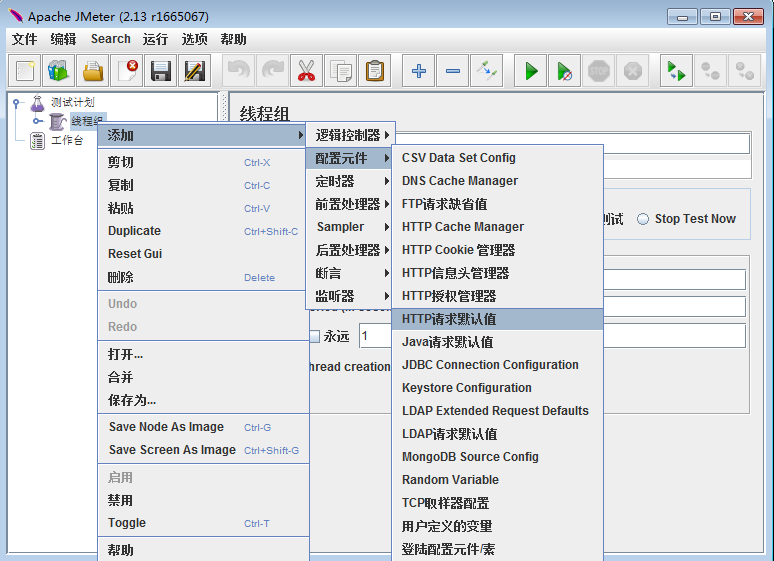
　　2、添加线程组：在“测试计划”上点击鼠标右键-->添加-->threads(Users)-->线程组，添加测试场景设置组件，接口测试中一般设置为1个“线程数”，根据测试数据的个数设定“循环次数”。



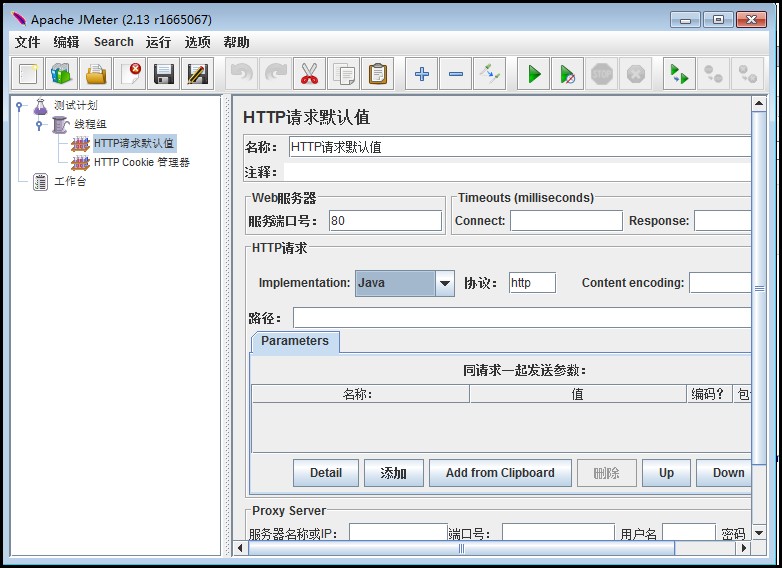
　　3、添加“HTTP Cookie管理器”：



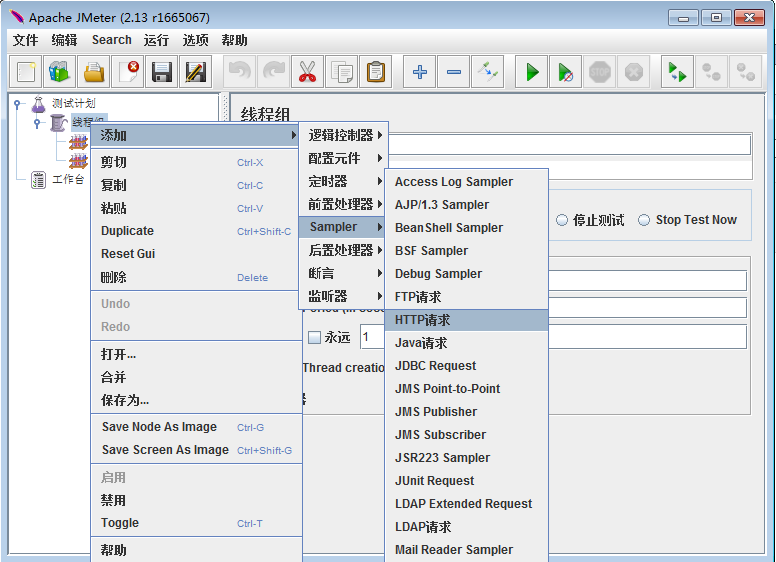
　　4、添加“Http请求默认值”组件，当被测系统有唯一的访问域名和端口时，这个组件很好用：



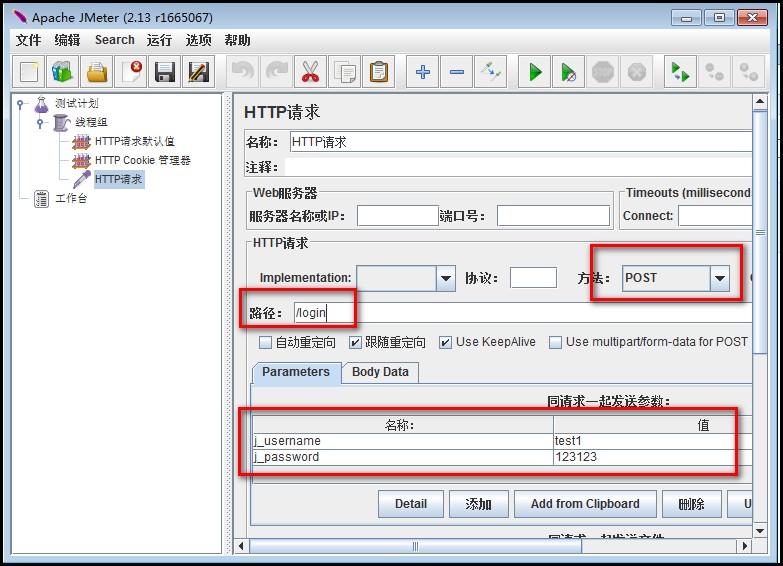
　　5、在“HTTP 请求默认值”组件配置页面，填写被测系统的域名和端口，http请求的实现包版本以及具体协议类型，线程组里的所有“HTTP Sampler”可默认使用此设置。



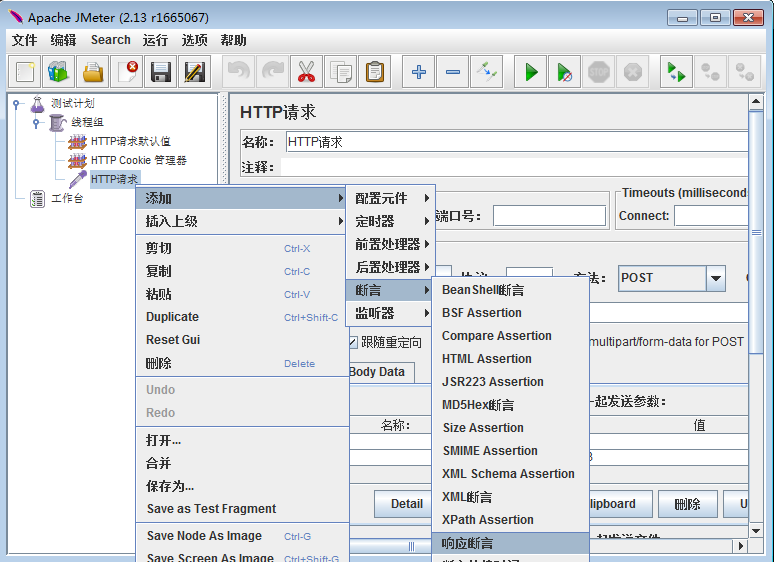
　　6、在“线程组”里添加“HTTP 请求”的Sampler



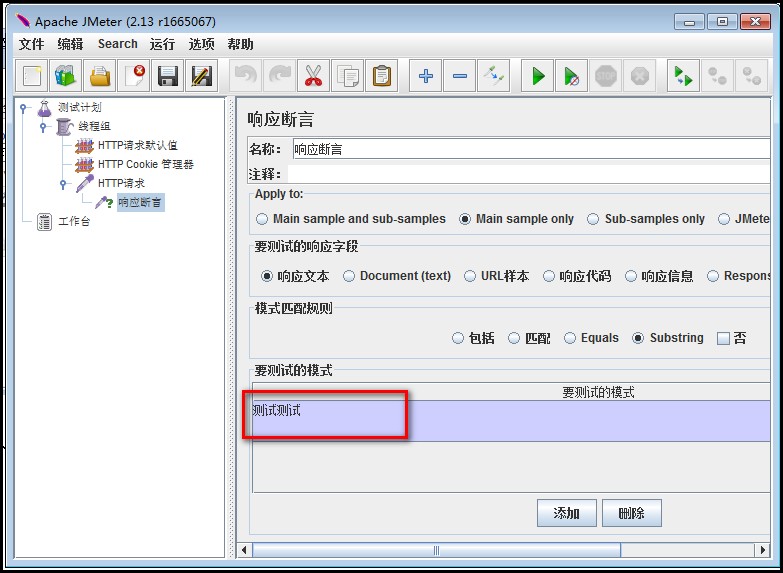
7、在HTTP请求设置页面，录入被测接口的详细信息，包括请求路径，对应的请求方法，以及随请求一起发送的参数列表：



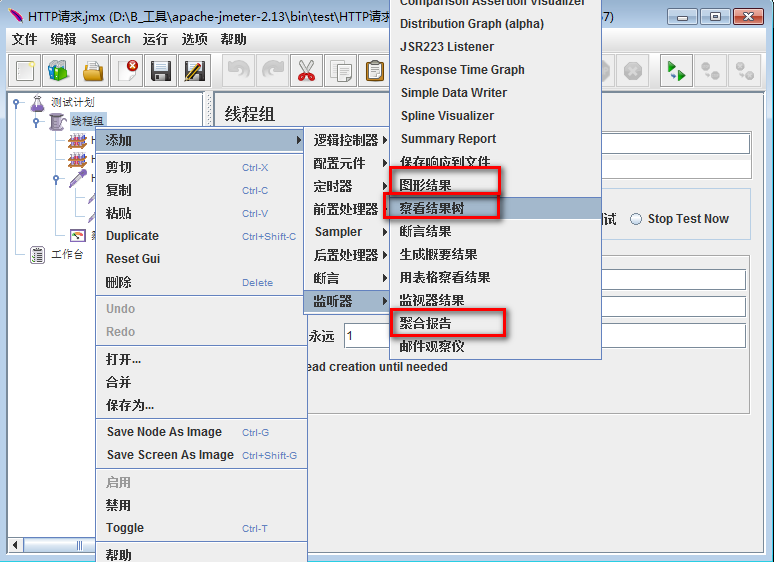
　　8、设置检查点：在被测接口对应的“HTTP 请求”上，添加“响应断言”。



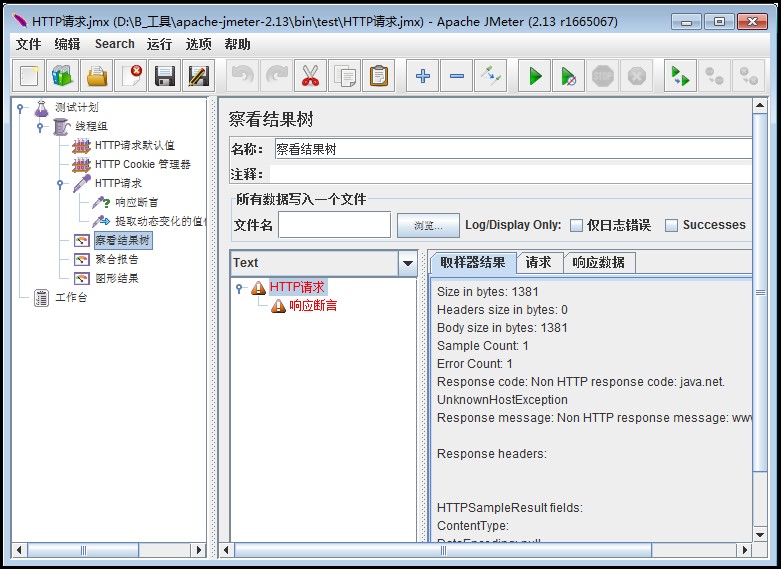
　　9、在设置页面上添加对相应结果的正则表达式存在性判断即可：



　　10、添加监听器：方便查看运行后的结果



　　运行结果：



　　上述步骤完成了一个简单测试案例的创建，复杂测试案例均在此基础上扩展完成。使用Jmeter工具开发的接口测试案例，一个子系统建议放在同一个 “测试计划”中，流程测试可以通过“线程组”来区分，这样也便于设定不同的测试数据个数。比较独立的接口，可以统一放在一个线程组内，顺序完成测试。

　　流程性接口的测试：如果要测试的接口可以组成一个流程，只需要顺序添加多个“HTTP 请求”的Sampler，各请求之间可以提取需要在上下文传递的数据作为参数，以保证流程中数据的一致性。