# 课程介绍

* 1. Jmeter核心知识（掌握）
* 2. 断言、函数、分布式（掌握）
* 3. 非GUI模式运行(掌握)

Jmeter核心知识

发送HTTP请求

启动jmeter，默认有一个测试计划，然后，修改计划名称，尽量使其变得有意义，容易看懂，然后，新建一个线程组，线程组设为1，方便演示。



配置元件HTTP属性管理器

另外，JMeter不是浏览器，因此其行为并不和浏览器完全一致。这些JMeter提供的配置元件中的HTTP属性管理器用于尽可能模拟浏览器行为，在HTTP协议层上发送给被测应用的http请求

**（1）HTTP Request Defaults（请求默认值）**

  用于设置其作用范围内的所有HTTP的默认值，可被设置的内容包括HTTP请求的host、端口、协议等

**（2）HTTP Authorization Manager（授权管理器）**

    用于设置自动对一些需要NTLM验证的页面进行认证和登录

**（3）HTTP Cache Manager**

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

**（4）HTTP Cookie Manager（cookie管理器）**

    用于管理Test Plan运行时的所有Cookie。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

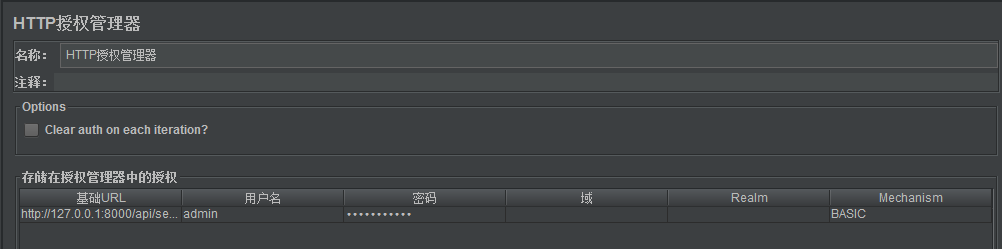
    同时，用户也可以在HTTP Cookie Manager中手工添加一些Cookie，这些被手工添加的Cookie会在发送请求时被自动附加到请求

**（5）HTTP Header Manager（信息头管理器）**

    用于定制Sampler发出的HTTP请求的请求头的内容。不同的浏览器发出的HTTP请求具有不同的Agent

    访问某些有防盗链的页面时需要正确的Refer...这些情况下都需要通过HTTP Header Manager来保证发送的HTTP请求是正确的

HTTP授权管理器内容如下填写：

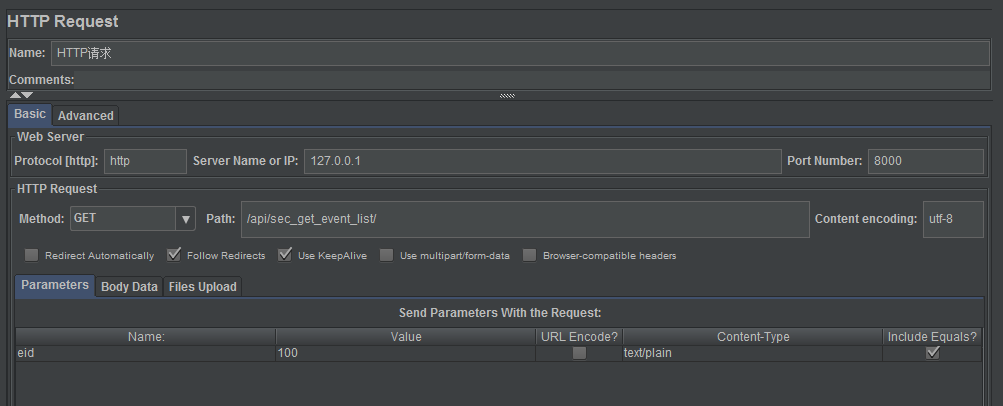


基础URL：填写被测地址

用户名：授权认证的用户名

密码：授权认证的密码

接着，添加Sampler（取样器）→http请求



按照截图，填入测试的服务器地址、端口、所用的户协议、方法，这里方法我用的是POST，然后填入路径，选择参数（parameters）；

HTTP请求的的属性参数说明

**1）名称：**用于标识一个sample。建议使用一个有意义的名称

**2）注释：**对于测试没任何影响，仅用来记录用户可读的注释信息

**3）服务器名称或IP：**http请求发送的目标服务器名称或者IP地址，比如http://www.baidu.com

**4）端口号：**目标服务器的端口号，默认值为80，可不填

**5）协议：**向目标服务器发送http请求时的协议，http/https，大小写不敏感，默认http

**6）方法：**发送http请求的方法(链接：http://www.cnblogs.com/imyalost/p/5630940.html）

**7）Content encoding：**内容的编码方式（Content-Type=application/json;charset=utf-8）

**8）路径：**目标的URL路径（不包括服务器地址和端口）

**9）自动重定向：**如果选中该项，发出的http请求得到响应是301/302，jmeter会重定向到新的界面

**10）Use keep Alive：**jmeter 和目标服务器之间使用 Keep-Alive方式进行HTTP通信（默认选中）

**11）Use multipart/from-data for HTTP POST ：**当发送HTTP POST 请求时，使用

**12）Parameters、Body Data以及Files Upload的区别：**

**1.** parameter是指函数定义中参数，而argument指的是函数调用时的实际参数

**2.** 简略描述为：parameter=形参(formal parameter)， argument=实参(actual parameter)

**3.**在不很严格的情况下，现在二者可以混用，一般用argument，而parameter则比较少用

   While defining method, variables passed in the method are called parameters.

   当定义方法时，传递到方法中的变量称为参数.

   While using those methods, values passed to those variables are called arguments.

   当调用方法时，传给变量的值称为引数.（有时argument被翻译为“引数“）

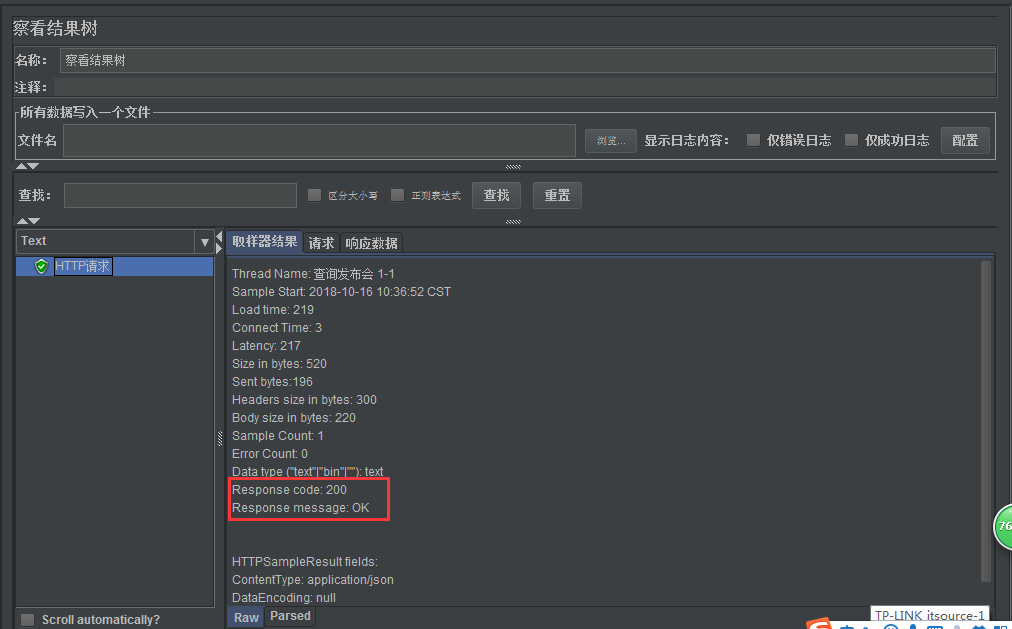
**4、**Body Data指的是实体数据，就是请求报文里面主体实体的内容，一般我们向服务器发送请求，携带的实体主体参数，可以写入这里（json格式的参数）

**5、**Files Upload指的是：从HTML文件获取所有有内含的资源：被选中时，发出HTTP请求并获得响应的HTML文件内容后还对该HTML进行Parse 并获取HTML中包含的所有资源（图片、flash等）：（默认不选中）

      如果用户只希望获取特定资源，可以在下方的Embedded URLs must match 文本框中填入需要下载的特定资源表达式，只有能匹配指定正则表达式的URL指向资源会被下载

注：如果请求参数是XML格式，则使用取样器中的SOAP/XML-RPC Request

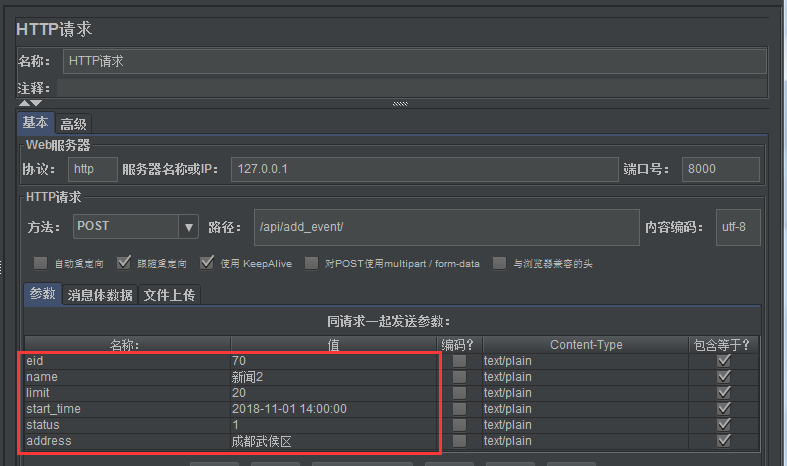
添加好监视器—查看结果树，点击运行，开始测试



参数化

为什么要参数化

现在有一个需求，添加100条学院信息，需要有100组不同的参数来一一请求



然而我们会发现，我们的键值对都是写死的，每次请求都需要更改，无法实现快速添加的需求。想要快速实现该需求，就需要用到参数化

什么是参数化

根据需求动态获取数据并进行赋值的过程

jmeter中参数化常用方式

**CSV Data Set Config：CSV数据控件**

**User Defined Variables：用户定义的变量**

**User Variables：用户参数**

jmeter中参数的定义

jmeter中参数化的变量用**${变量名}**表示

CSV Data Set Config

一种从外部读取数据功能的组件

**适用范围**：适用于需要大量测试数据的前提，如给系统增加100条数据，设置 1000名用户登录等等

**实现步骤:**

1. 创建测试计划

2. 创建线程组

3. 在测试计划里面添加CSV Data Set Config组件

4. 在线程组里面添加http请求

5. 添加察看结果树组件 查看结果

**组件要点分析：**

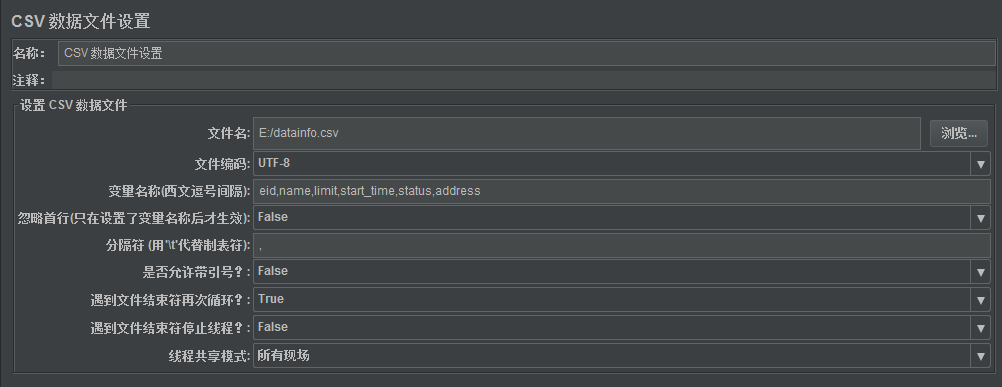
1. 线程组:如果添加10条数据,线程组属性循环次数就设置为10

2. CSV Data Set Config 读取变量配置

3. HTTP请求：parameters填写(参数) 方法(POST)

4. 参数化引用格式：${参数名} 如：${dep\_id}

**CSV Data Set Config配置**

****

**说明：**

**Filename（文件名）：**E:\datainf0.csv文件名，保存参数化数据的文件目录，可选择相对或者绝对路径（建议填写相对路径，避免脚本迁移时需要修改路径）;

**File encoding:**UTF-8(文件编码)，E:\datainf0.csv文件的编码格式，在保存时保存编码格式为UTF-8即可;

**Variable Names**(comma-delimited)（变量名称）：对对应参数文件每列的变量名，类似excel文件的文件头，起到标示作用，同时也是后续引用的标识符，建议采用有意义的英文标示;

(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，这里的eid,name…可以被利用变量名来引用：${eid},${name};

Lgnore first line（忽略首行）：如果首行写的是变量名，则选择true

**Delimitet：**参数文件分隔符，用来在“Variable Names”中分隔参数，与参数文件中的分隔符保持一致即可; 如文件中使用的是逗号分隔，则填写逗号；如使用的是TAB，则填写\t;

**Allow quote data：**是否允许引用数据，默认false，选项选为“true”的时候对全角字符的处理出现乱码 ;

**Recycle on EOF？：**是否循环读取参数文件内容；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;

△ Ture：为true时，当已读取完参数文件内的测试用例数据，还需继续获取用例数据时，此时会循环读取参数文件数据（即：读取文件到结尾时，再重头读取文件）;

**△**False：为false时，若已至文件末尾，则不再继续读取测试数据；通常在“线程组线程数\* 线程组循环次数＞参数文件行数”时，选用false（即：读取文件到结尾时，停止读取文件）;

**Stop thread on EOF?：**当Recycle on EOF为False时（读取文件到结尾），停止进程，当Recycle on EOF为True时，此项无意义;

**△**若为ture，则在读取到参数文件行末尾时，终止参数文件读取线程；

**△**若为false，此时线程继续读取，但会请求错误，因此时读取的数据为EOF;

**Sharing mode:**共享模式，即参数文件的作用域，有以下几种方式：

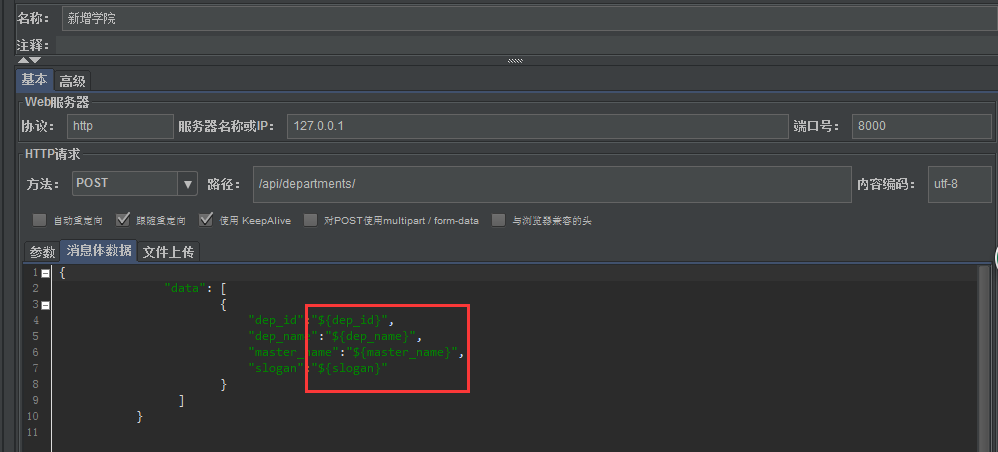
**△**All threads:当前测试计划中的所有线程中的所有的线程都有效，默认;

**△**Current thread group:当前线程组中的线程有效;

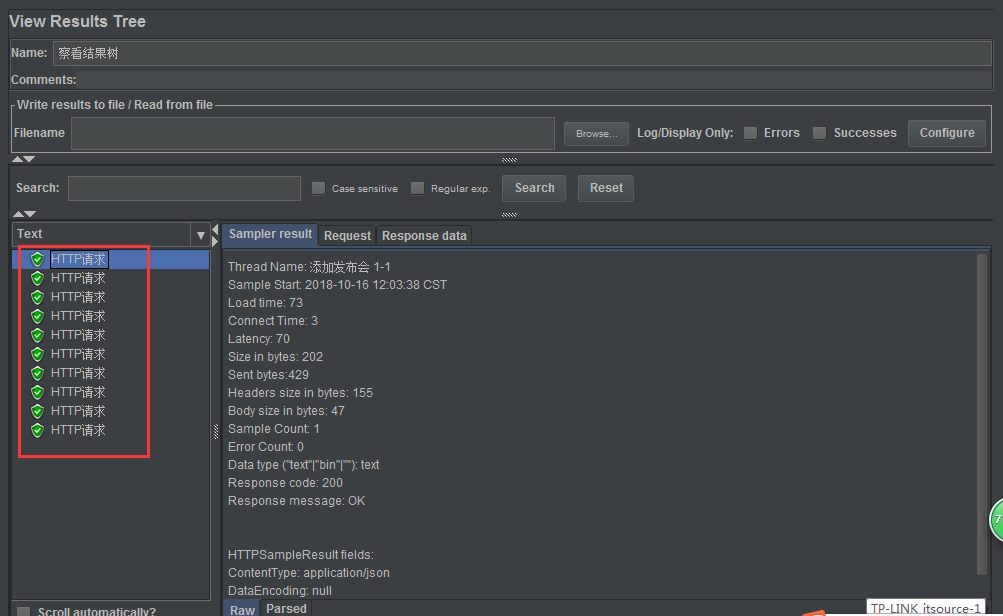
**△**Current thread:当前线程有效;

**HTTP请求中的设置**

参数化后，请求参数的值直接使用${变量名代替}，如图：

****

添加结果树，运行查看结果，10条数据添加成功

****

****User Variables：用户参数****

**适用范围：**适用于参数取值范围很小的时候使用

**用户参数实现的步骤分析：**

1.创建测试用例

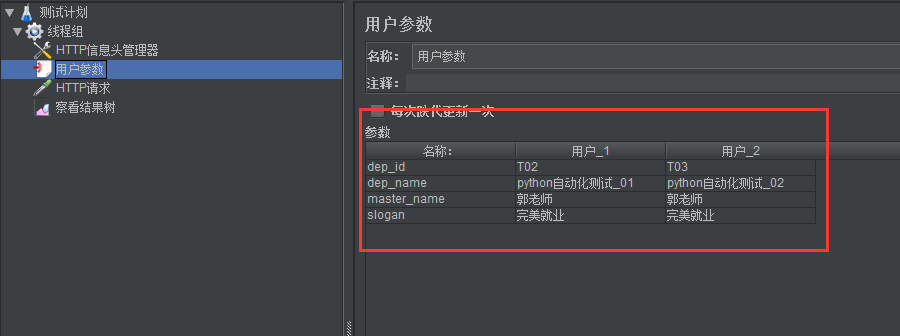
2.在测试用例里面添加http请求(实现学生管理系统新增一条记录)

3.把要提交的数据定义到用户参数组件里面（前置处理器——用户参数）

4.定义好数据后和http请求组件关联

5.创建察看结果树组件,运行并查看数据库是否添加成功

用户参数设置：



如上图所示，在该参数组中已经设置了四个参数，dep\_id,dep\_name..分别有2组不同的取值，通过页面下方的四个按钮，可以增加删除参数的可能取值。

PS：User Variables中设置的参数可以在test plan执行过程中发生变化。

添加结果树运行脚本

****User Defined Variables：用户定义的变量****

**适用范围**：适用于测试过程中的常量配置，如数据库地址，测试环境、开发环境地址等常量配置

**实现步骤分析：**

创建测试计划

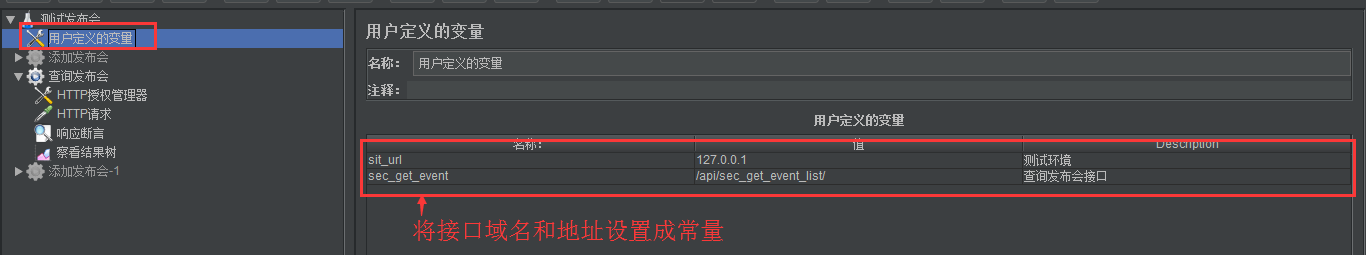
配置元件——用户定义的变量

添加线程组

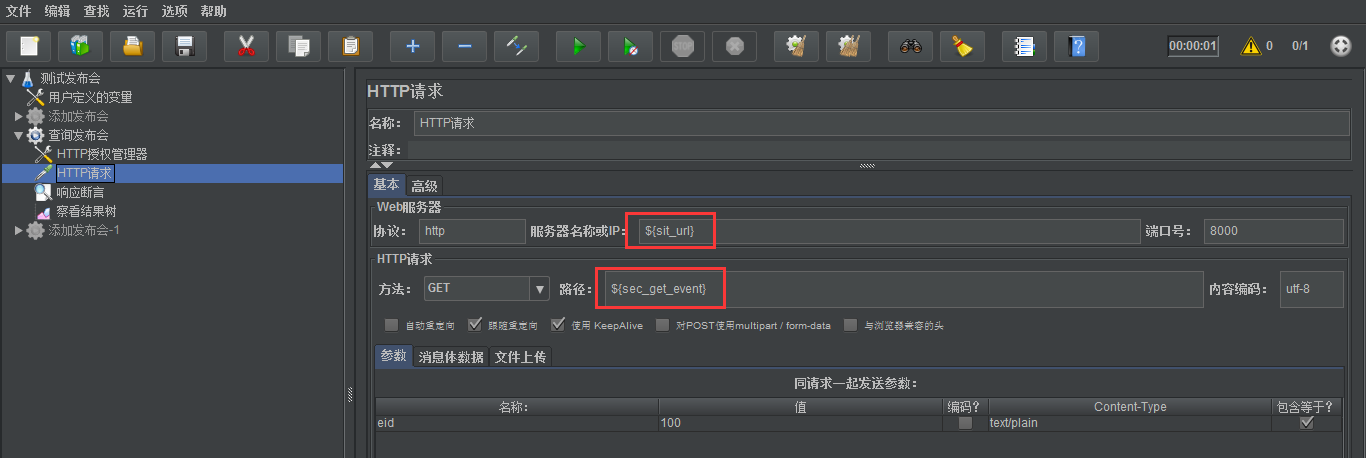
添加http请求

添加结果树

用户定义的变量配置：



调用常量：



Jmeter连接数据库（MySQL）

为什么要使用jmeter连接数据库

我们测试的软件项目所有的数据都存储在数据库,直接链接数据库把数据取出来和通过接口的方式取出来,如果数据一致,说明后台程序开发的没有问题.

jmeter如何连接数据库

由于jmeter本身没有提供链接数据库的功能,所以我们需要通过三方的工具. 下载mysql驱动，根据自己mysql的版本下载对应驱动，将驱动放在jmeter的bin目录下。

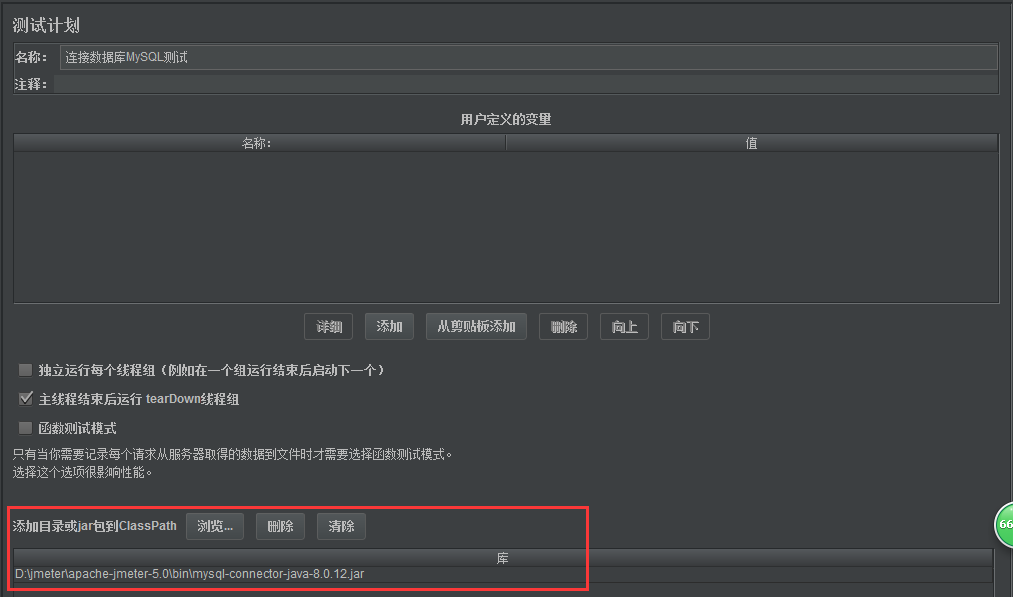
mysql各种版本驱动jar包下载地址:

http://central.maven.org/maven2/mysql/mysql-connector-java/

连接数据库的步骤

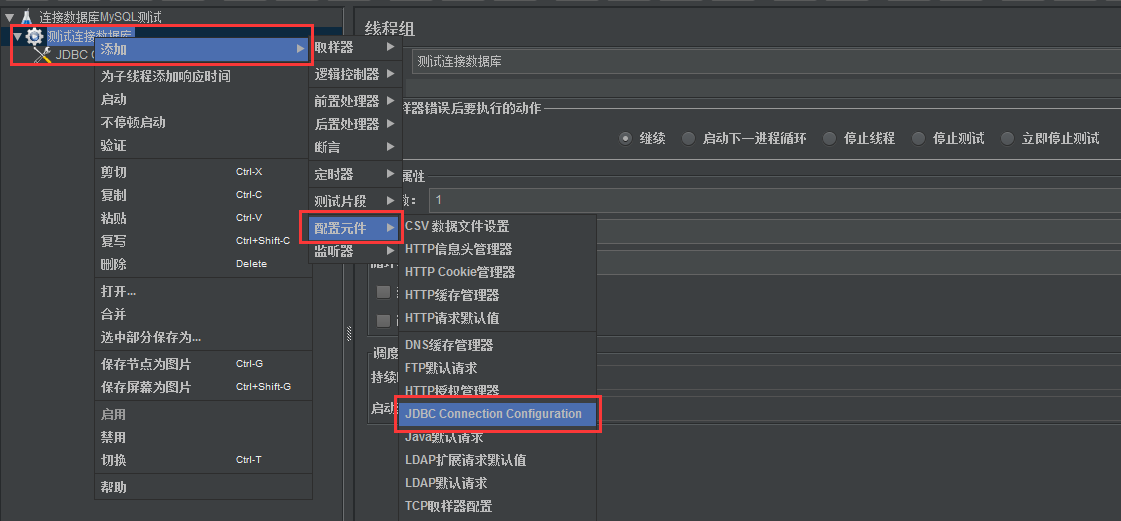
1.创建测试计划（确保已下载对应版本的mysql驱动）

2.在测试计划里面添加依赖(添加需要的jar包)

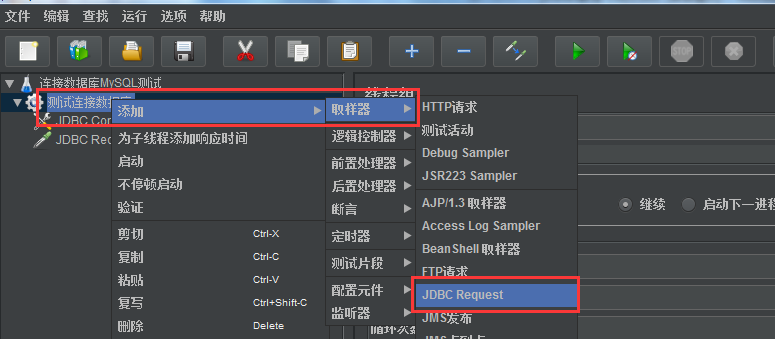


3.创建线程组

4.在线程组里面添加一个jdbc connection configuration组件



5.添加一个专门用于向数据库发送sql语句的组件(JDBC Request)



jdbc connection configuration组件参数详解



**Variable Name（变量名）：**这里写入数据库连接池的名字

**Database URL：**数据库连接地址，如：jdbc:mysql:// 数据库IP地址:数据库端口/数据库名称

**JDBC Driver class：**数据库驱动

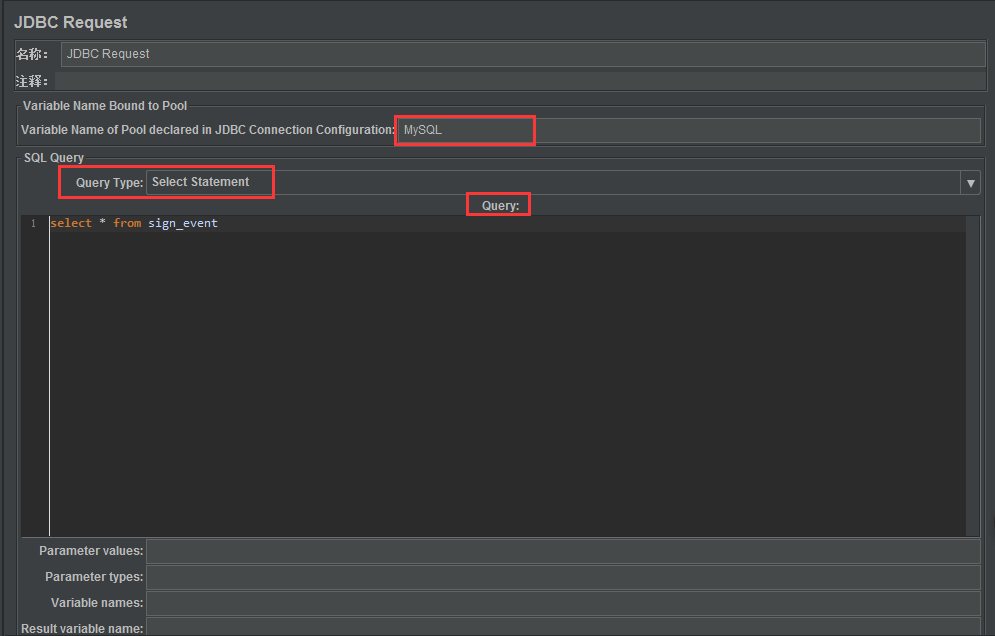
**Username：**数据库登录名

**Password：**数据库登陆密码

这里顺带说说不同数据库的驱动类和URL格式：

|  |  |  |
| --- | --- | --- |
| Datebase | Driver Class | Datebase URL |
| MySQL | com.mysql.jdbc.Driver | jdbc:mysql://host:port/dbname |
| SQLServer | com.microsoft.sqlserver.jdbc.SQLServerDriver  或者  net.sourceforge.jtds.jdbc.Driver | jdbc:sqlserver://host:port;databaseName=dbname |
| Oracle | oracle.jdbc.driver.OracleDriver | jdbc:oracle:thin:@//host:port/service  jdbc:oracle:thin:@(description=(host={mc-name})  (portocol=tcp)(port=[port-no]))  (connect\_data=(sid={sid}))) |
| PostgreSQL | org.postgresql.Driver | jdbc:postgresql:{dbname} |
| Ingres（2006） | ingres.jdbc.IngresDriver | jdbc:ingres://host:post/db[;attr=value] |

JDBC Request



**Variable name：**这里写入数据库连接池的名字（和JDBC Connection Configuration名字保持一致 ）

**Query：**里面填入查询数据库数据的SQL语句（填写的SQL语句末尾不要加“；”）

**parameter valus：**数据的参数值

**parameter types：**数据的参数类型

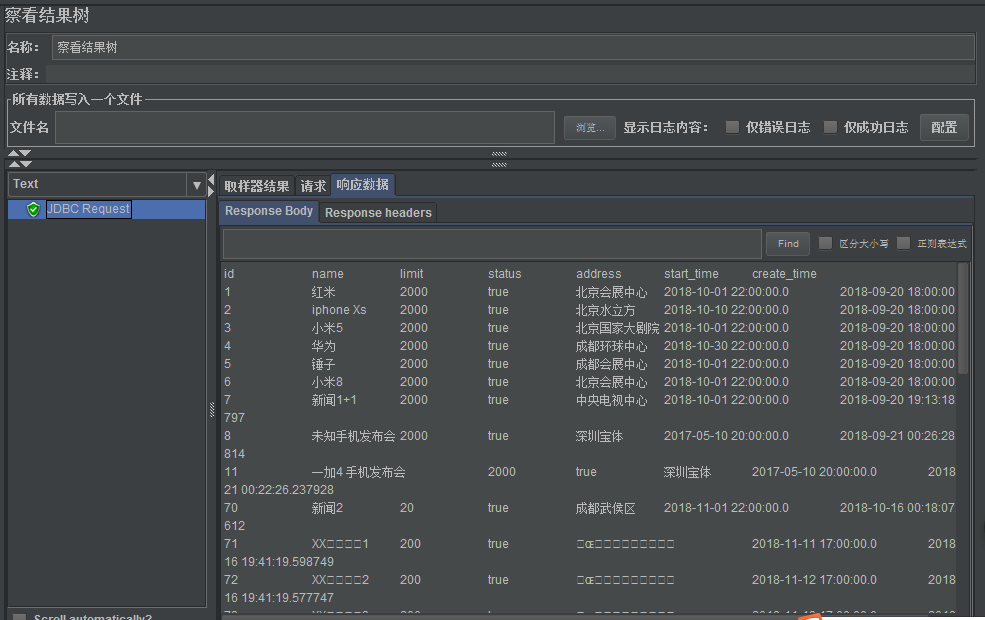
**variable names：**保存SQL语句返回结果的变量名

**result variable name：**创建一个对象变量，保存所有返回结果

**query timeout：**查询超时时间

**handle result set：**定义如何处理由callable statements语句返回的结果

添加结果树查看执行结果：



注：

当查询（select）的时候 使用  select Statement

当新增（insert）、删除（delete）、更新（update）都使用 update Statament

The server time zone value 异常处理

Cannot create PoolableConnectionFactory (The server time zone value '???��������??��??' is unrecognized or represents more than one time zone. You must configure either the server or JDBC driver (via the serverTimezone configuration property) to use a more specifc time zone value if you want to utilize time zone support.)

解决方法：数据库连接url后面加 **serverTimezone=GMT**或**serverTimezone=GMT%2B8**

执行多条SQL语句时

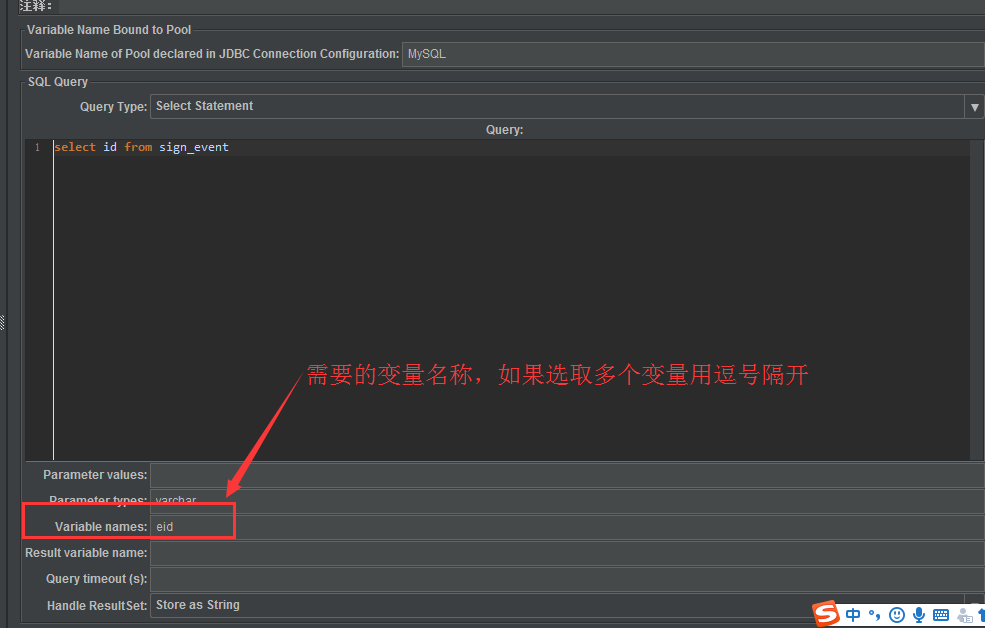
查询语句select 和update、insert 语句不能在一个JDBC request 中执行；

当执行多条SQL 语句时，每条语句后面加; 并且在 Database URL 后增加一个参数**allowMultiQueries=true**

Datebase URL后增加参数，在dbname后加”?”，如有多个参数，每个参数用”&”隔开，如：

jdbc:mysql://ecshop.itsoso.cn:3306/ecshop?serverTimezone=GMT&useUnicode=true&characterEncoding=utf-8&allowMultiQueries=true&zeroDateTimeBehavior=CONVERT\_TO\_NULL

数据库查询参数获取



变量的选取，${eid\_1}选取第一个变量。查看变量使用debug sampler中查看

关联(重点)

关联的定义

从上一条请求中获取数据，使用在下一条请求中的过程。

需求1

访问网易官网，获取title值

将title值 放入百度搜索框

实现需求的思路

1.创建测试计划

2.创建线程组

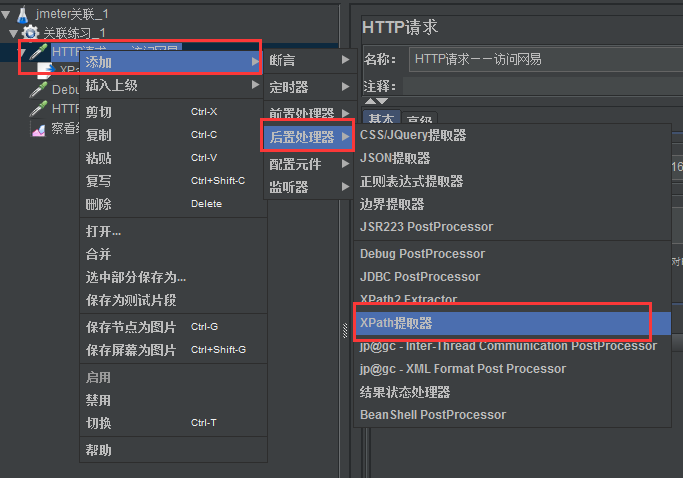
3.在线程组里面创建2个http请求 第一个请求网易官网

4.通过xpath提取器获取title的值 把获取到的值赋值给一个变量mytitle

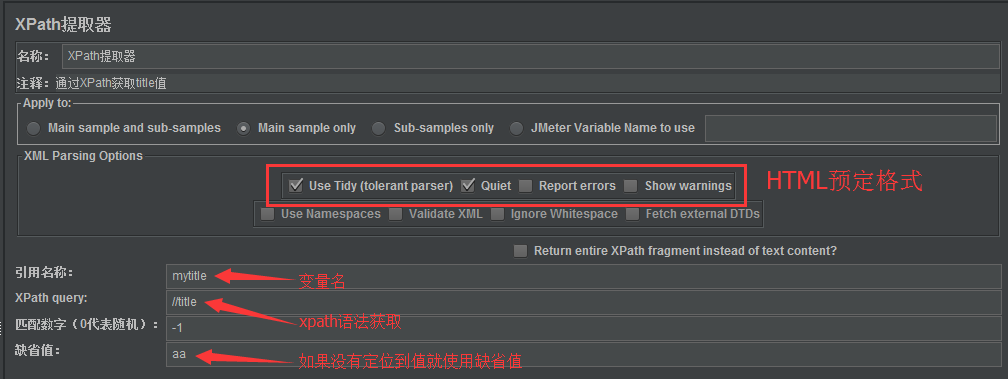
5.使用获取到的值访问百度

Xpath提取器

添加Xpath提取器：



Xpath提取器的配置



需求2

用户登录系统并在系统中查询车辆信息

实现思路：

1.创建测试计划

2.创建2个http请求

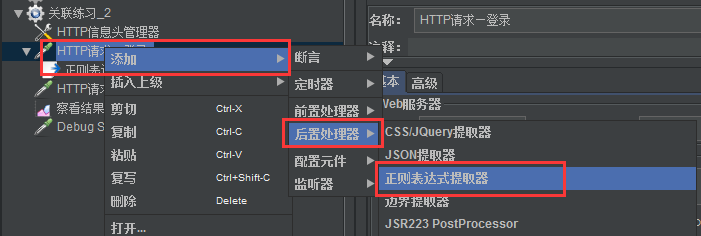
3.第一个http请求登录

4.通过正则表达式把token取出来

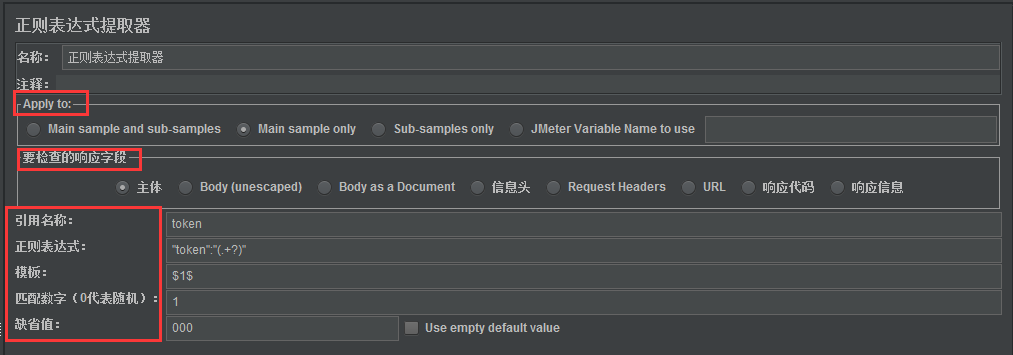
5.根据第一步取出来的token,传入查询车辆的接口中

正则表达式提取器

添加正则表达式提取器



正则表达式提取器配置



**后置处理器：**在请求结束或者返回响应结果时发挥作用

**正则表达式提取器：**允许用户从服务器的响应中通过使用正则表达式提取值。该元素会作用在指定范围取样器，用正则表达式提取所需值，生成模板字符串，并将结果存储到给定的变量名中。

**APPly to:**作用范围（返回内容的断言范围）

         Main sample and sub-samples:作用于父节点的取样器及对应子节点的取样器

         Main sample only：仅作用于父节点的取样器

         Sub-samples only:仅作用于子节点的取样器

         JMeter Variable:作用于jmeter变量(输入框内可输入jmeter的变量名称)

**要检查的响应字段：**需要检查的响应报文的范围

         主体：响应报文的主体

         Body(unescaped):主体，响应的主体内容且替换了所有的html转义符，注意html转义符处理时不考虑上下文，因此可能有不正确的转换，不太建议使用

         Body as a Document：从不同类型的文件中提取文本，注意这个选项比较影响性能

         Response Headers：响应信息头

         Request Headers:请求信息头

         URL：统一资源定位符，即Internet上用来描述信息资源的字符串

         Response Code:响应状态码，比如200、404等

         Response Message:响应信息

**PS：**jmeter的中文翻译有时候不太准确，建议尽量选择语言格式为英文（为了方便说明，这里选择中文语言，当然，自己明白最好，不用纠结这个）

**引用名称（Reference Name）：**Jmeter变量的名称，存储提取的结果；即下个请求需要引用的值、字段、变量名（例子中我提取的是SOCIAL\_NO）

**引用方法：**引用方法:${引用名称}

**正则表达式（Regular Expression）**：使用正则表达式解析响应结果，**“（）”表示提取字符串中的部分值**，请不要使用“||”，除非你本身需要匹配这个字符。

 . 代表匹配任意一个字符

   [] 比如 [0-9]代表:匹配0-9之间任意一个数字

              [a-z]代表:匹配a-z之间任意一个字符

              [A-Z]代表:匹配A-Z之间任意的一个字符

  +  匹配一次或多次

  ?  代表匹配一次或一次也没有   这个符号还有特殊的用法. 当放到\*号后面的时候 标识取到的数据是非贪婪的.

  \*  零次 或多次

   \d:  数字 [0-9]

  \w: [a-zA-Z\_0-9]

    \b[a-z]{3}\b :\b代表边界

  {3}:代表匹配3次

 [1-9][0-9]{4,14} :代表 至少匹配4次,最多不超过14次

  1[358]\d{9} 匹配手机号

  [a-zA-Z0-9\_]+@[a-zA-Z0-9]+\.[a-zA-Z]+ 匹配邮箱.

常用正则表达式查询:<http://tool.oschina.net/uploads/apidocs/jquery/regexp.html>

**模板（Template）：**模板，用来从匹配的结果中创建一个字符串，这是通过正则表达式匹配出来的一组值，语法为：$1$指代第一组，$2$指代第二组，$0$指代整个匹配结果

**匹配数字（Match No）：**正则表达式匹配数据的结果可以看做一个数组，表示如何取值：0代表随机取值，正数n则表示取第n个值（比如1代表取第一个值），负数则表示提取所有符合条件的值。

**缺省值：**匹配失败时候的默认值；通常用于后续的逻辑判断，一般通常为特定含义的英文大写组合，比如：ERROR

正则与Xpath的区别

正则表达式提取器和XPath Extractor的区别：

正则表达式提取器可以用于对页面任何文本的提取，提取的内容是根据正则表达式在页面内容中进行文本匹配；

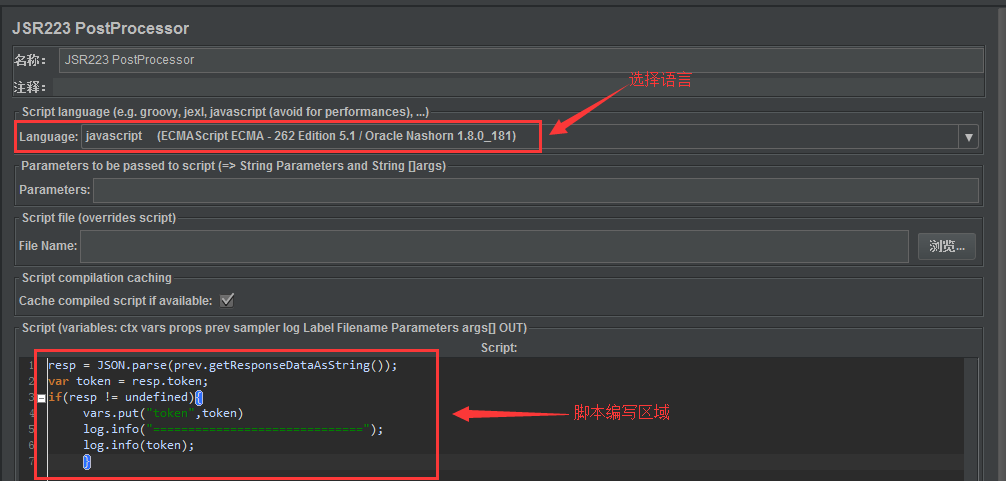
XPath Extractor则可以提取返回页面任意元素的任意属性；

如果需要提取的文本是页面上某元素的属性值，建议使用XPath Extractor;

如果需要提取的文本在页面上的位置不固定，或者不是元素的属性，建议使用正则表达式提取器。

通过脚本提取关联值

在jmeter中除了上述2中方法外，还可以通过语言脚本来提取所需要的关联值，如JSR223 PostProcessor，在该提取器中，支持JavaScript、Java等语言。如下图，使用的是JavaScript脚本语言编写提取返回值中的token值。



response = JSON.parse(prev.getResponseDataAsString());

var uid = response.data.session.uid;

var sid = response.data.session.sid;

if(response != undefined){

vars.put("uid",uid)//将uid转换成jmeter的变量

vars.put("sid",sid)

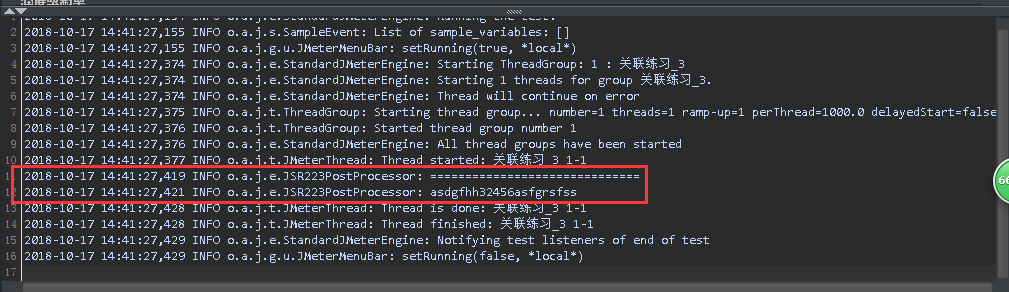
log.info("++++++++++++++++++++++++++++++++++++++++++++++")

log.info(uid)

log.info(sid)

}

运行脚本时一定要与HTTP请求组合，否则无法单独运行，查看脚本运行结果点击jmeter界面的右上角” ”查看。



断言、集合点、函数、分布式

断言

定义

断言就是让程序代替人工去判断程序响应数据是否达到预期结果

为什么学习断言

断言可以检查返回数据的正确性与完整性

常见的断言

响应断言 (断言状态码 + 断言的内容)

长度断言 (断言响应内容的长度)

持续时间断言 (断言响应的时间)

响应断言的使用（掌握）

需求：判断登录返回的状态码是否为200

实现步骤

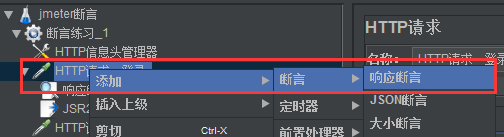
1.创建测试计划

2.创建一个http请求访问登录接口

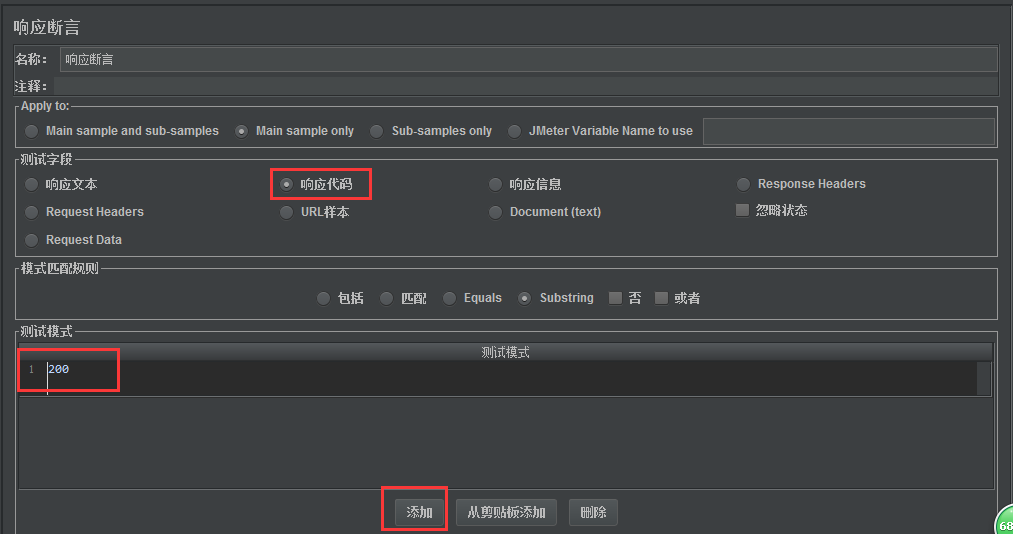
3.在http请求里面添加响应断言组件

4.创建察看结果树组件 察看结果

添加响应断言组件



配置响应断言



注意:添加断言组件后,如果添加的断言正确,结果树里面没有提示.

断言返回值为中文,如果直接断言中文不成功,是因为返回的中文被jmeter进行了Unicode转码,需要在对应的请求后先添加Bean Shell PostProcessor后置处理器,在该处理器中加入如下代码,即可直接断言中文

//获取响应代码Unicode编码的

String s2=new String(prev.getResponseData(),"UTF-8");//---------------一下步骤为转码过程---------------

char aChar;

int len= s2.length();

StringBuffer outBuffer=new StringBuffer(len);

for(int x =0; x <len;){

aChar= s2.charAt(x++);

if(aChar=='\\'){

aChar= s2.charAt(x++);

if(aChar=='u'){

int value =0;

for(int i=0;i<4;i++){

aChar= s2.charAt(x++);

switch(aChar){

case'0':

case'1':

case'2':

case'3':

case'4':

case'5':

case'6':

case'7':

case'8':

case'9':

value=(value <<4)+aChar-'0';

break;

case'a':

case'b':

case'c':

case'd':

case'e':

case'f':

value=(value <<4)+10+aChar-'a';

break;

case'A':

case'B':

case'C':

case'D':

case'E':

case'F':

value=(value <<4)+10+aChar-'A';

break;

default:

throw new IllegalArgumentException(

"Malformed \\uxxxx encoding.");}}

outBuffer.append((char) value);}else{

if(aChar=='t')

aChar='\t';

else if(aChar=='r')

aChar='\r';

else if(aChar=='n')

aChar='\n';

else if(aChar=='f')

aChar='\f';

outBuffer.append(aChar);}}else

outBuffer.append(aChar);}//-----------------以上内容为转码过程---------------------------//将转成中文的响应结果在查看结果树中显示

prev.setResponseData(outBuffer.toString());

大小断言 （了解）

作用：主要判断返回数据的大小是否属于预期数据大小范围

实现:直接添加长度断言即可.

持续时间断言（了解）

作用：断言服务器响应请求的时间是否小于指定值

实现:直接添加时间断言组件察看

集合点（掌握）

集合点

作用：集合点用以同步虚拟用户，以便恰好在同一时刻执行任务。

需求

有50个用户同时查询同一个学院信息

解决方案分析

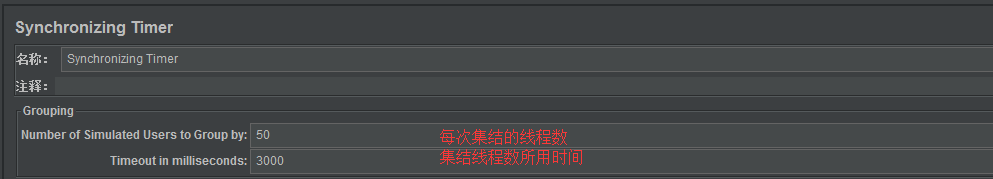
1.测试计划->线程组

2.线程组->Sampler->HTTP请求(查询学院)

4.HTTP请求(查询学院)->定时器->Synchronizing Timer

5.测试计划->监听器->察看结果树、聚合报告

集合点参数配置图



1. Number of Simulated Users to Group by:集合多少用户后再执行请求（也就是执行的线程数）

扩展选项：

2. Timeout in milliseconds：超时时间(毫秒)

1) 设置为0，无限等待，直到达到集合点设置的线程数。

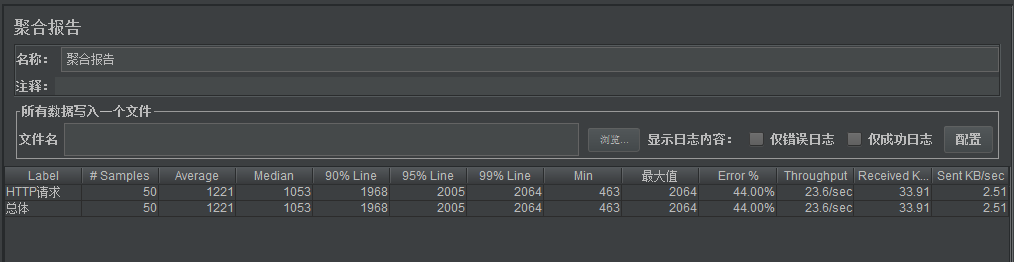
2) 设置指定时长，如果到达指定时长，集合点数量未到达，集合多少用户释放多少用户数量。

3. 细节:

1.总用户数量最好能整除一组用户数,否则在超时间为0时候,可能会导致程序挂起

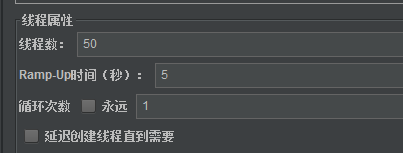
2.如果总用户数不能被一组用户数整除,建议超时时间不为0.

查看执行结果

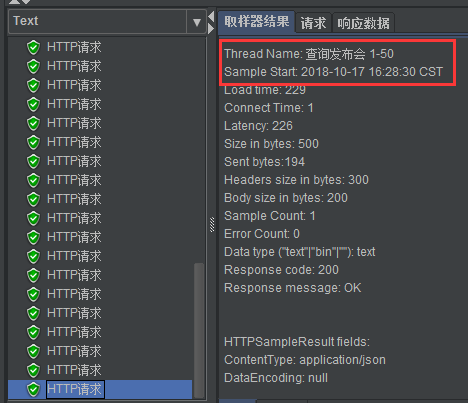
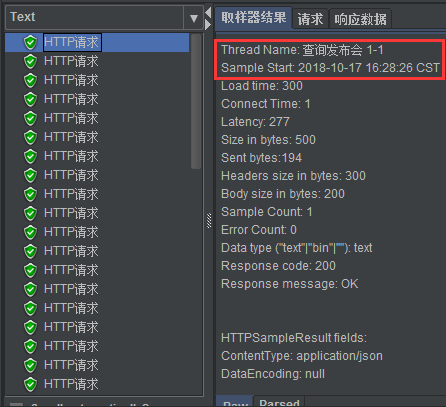


线程数与集合点的区别

线程数



表示5秒内启动50个线程，运行结果如下：

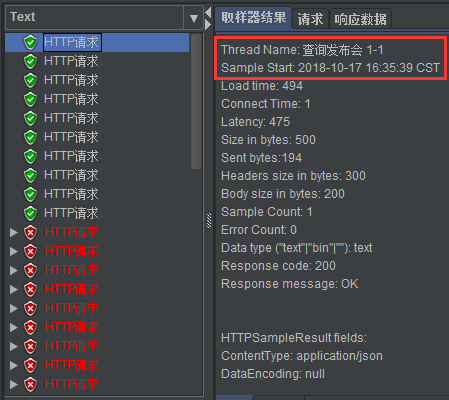
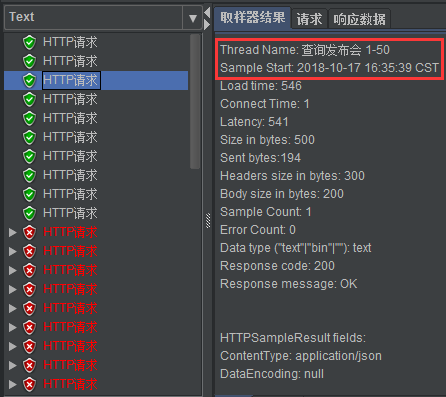


集合点

Number of Simulated Users to Group by：50，表示集合50个线程后再并发（该数值不能大于线程组的最大数字）

Timeout in milliseconds：10000，表示集合等待时间为10秒（该时间要大于线程组的启动时间，否则超时后就不再等待，直接并发。）

运行结果：线程1和线程50同时运行

函数

函数的分类

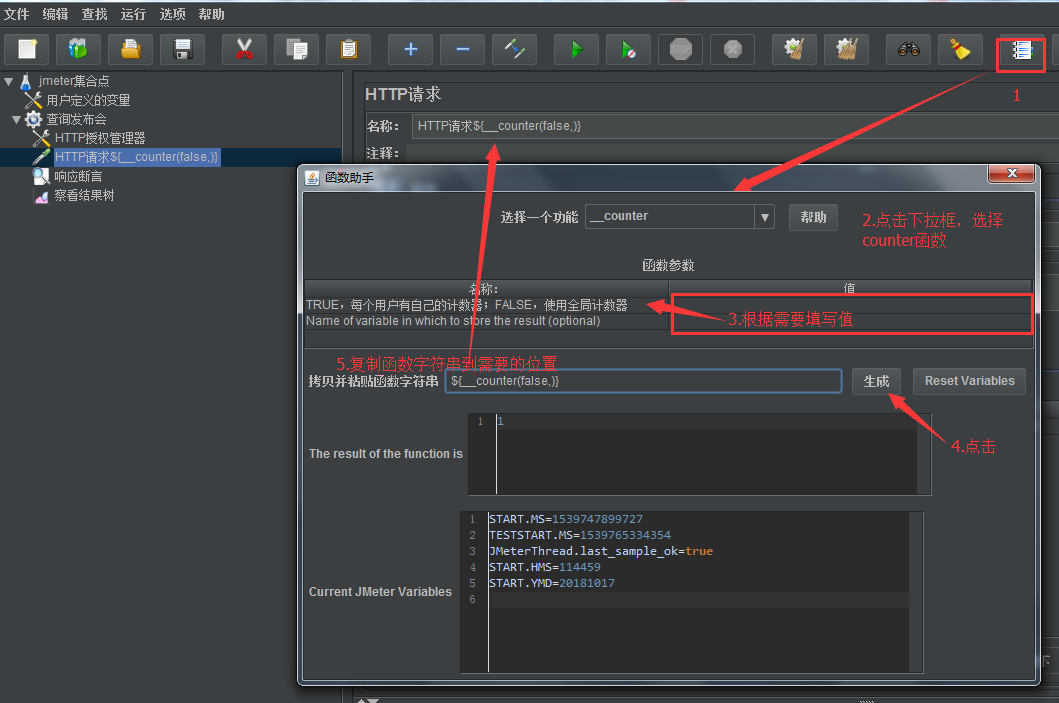
数字相关的函数

[1]\_counter计数器函数

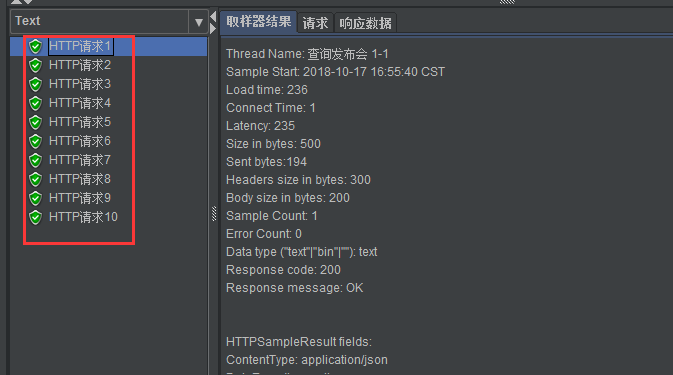
作用：这个函数是一个计数器，用于统计函数的使用次数

需求:

使用counter计次函数，请求(学院查询)10次,为每次请求打上标识.



执行结果：



注：

参数：true：每个用户都使用自己的计数器

false:代表全局计数器 所有的用户共用一个计数器

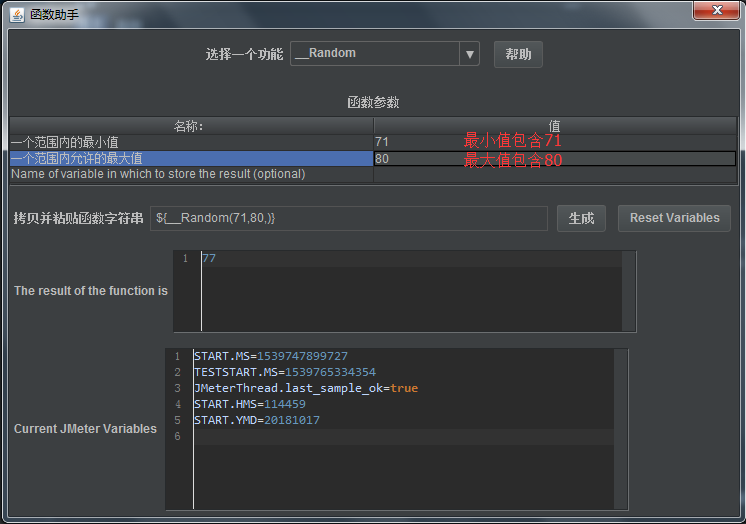
拷贝生成的函数,哪里需要就拷贝到哪里

[2]\_random();随机数函数

作用：随机生成一个指定范围内的数值

需求:

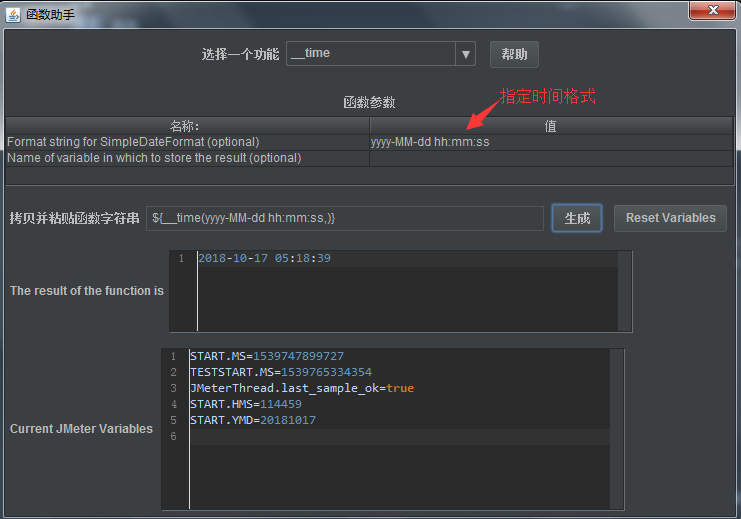
使用\_\_Random函数对请求(查询发布会)进行随机查询



[3]\_time();

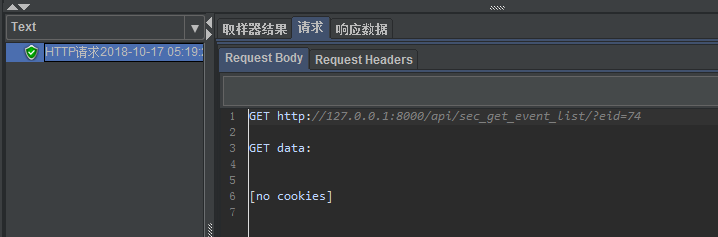
作用：获取当前系统的时间函数

生成时间格式:yyyy-MM-dd hh:mm:ss





运行结果：



关联相关的函数

[1] setProperty()函数和property()函数

实现跨线程组关联

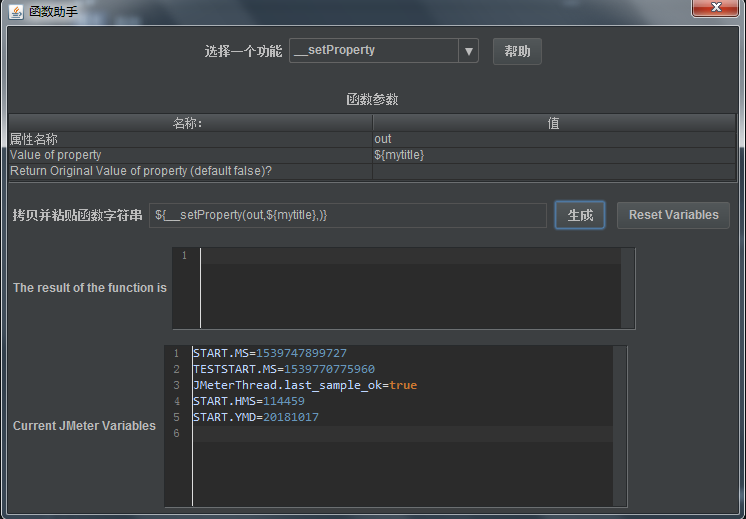
 实现的步骤

   1.创建测试计划 --->添加两个线程组

   2.在第一个线程组里面添加一个http请求 访问网易 通过xpath把title的值取出来

   3.在另外一个线程组里面添加http请求访问百度

   4.由于我们声明的mytitle变量只在线程组1里面有效 ,需要我们提升作用域.通过setProperty函数来提升



5.生成这个函数后 这个函数比较特殊 他必须的放到BeanShell Sampler组件里面  当把这个函数放到这个组件里面后 就相当于把变量mytitle的值放到了我们声明的公共空间里面

  6.其他线程组通过property函数去取数据 ${\_\_property(out,,)}

  7.创建结果树组件 察看结果.



分布式

什么是分布式

由多台电脑共同完成同一1个任务(请求)部署，我们称这种部署为分布式部署

为什么要使用分布式

由于一台电脑的CPU、内存有限，无法满足更高的测试要求

需要对服务器进行性能或压力测试时

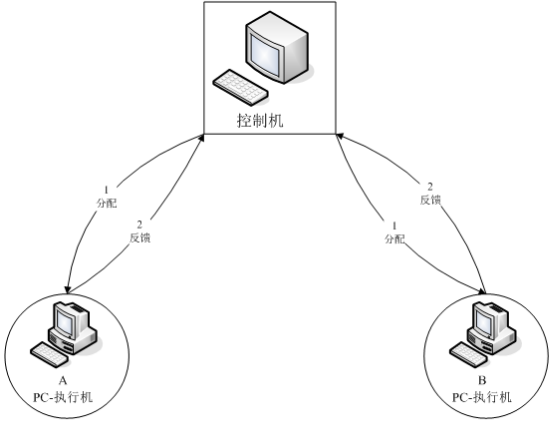
分布式原理

1. 一台电脑作为控制机(Controller、master)，其它电脑做为执行机(Agent、slave)

2. 执行时，控制机会把脚本发送到每台执行机上，执行机拿到脚本后就开始执行

3. 执行机执行时不需要启动Jmeter界面，可以理解它是通过命令行模式执行的

4. 执行完成后，执行机会把结果回传给控制机，控制机会收集所有执行机的信息并汇总



解决方案分析

1. 1台电脑(控制机)分发执行任务

2. 2台电脑(执行机)执行任务

3. 在执行机上启动监听服务程序

4. 在控制机上启动运行

5. 测试计划->聚合报告

技术难点分析

1. 执行机-jmeter.properties设置

2. 控制机jmeter.properties设置

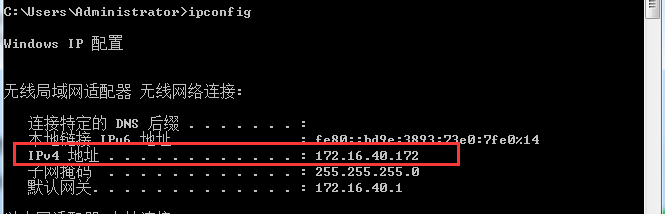
3. 执行机启动分布式监听服务程序

4. 2台执行机,用户数如何设置

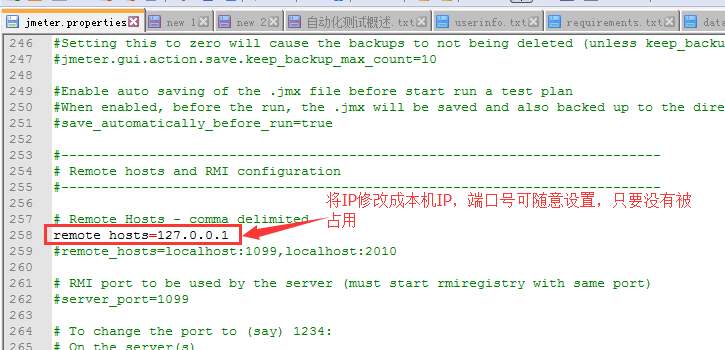
执行机配置

安装jmeter（略）

打开“运行”，输入"cmd",打开运行面板，输入“ipconfig”，找到IP地址，例如是：172.16.40.172



打开Jmeter/bin/jmeter.properties，找到”remote\_hosts=127.0.0.1”,把这一行修改为”remote\_hosts=172.16.40.172：1099,1099是端口号，可以随意自定义。



打开jmeter-server.bat文件，就设置完成了，等待控制机(Controller)启动。

控制机配置

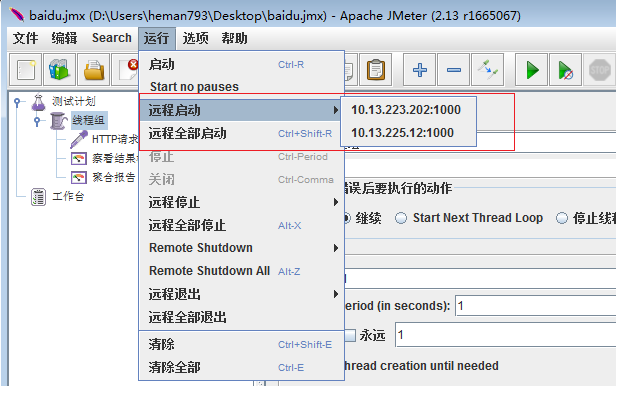
打开Jmeter/bin/jmeter.properties，找到”remote\_hosts=127.0.0.1”,把这一行修改为”remote\_hosts=172.16.40.172：1099。如果有多台控制机，这里需要把所有的控制机的IP地址和端口号都加入进来并用“,”隔开。

打开jmeter-server.bat文件，如果控制机不执行脚本，就不用打开。

执行脚本

配置线程数

运行——远程启动：可单选一个执行机执行，也可选择全部启动



注意事项

1、保持Contorller和Agent机器的JDK、jmeter以及插件等配置版本一致；

2、如果测试数据有用到CSV或者其他方式进行参数化，需要将data pools在每台Agent上复制一份，且读取路径必须保持一致；

3、确保Contorller和Agent机器在同一个子网里面；

4、检查防火墙是否被关闭，端口是否被占用（防火墙会影响脚本执行和测试结构收集，端口占用会导致Agent机报错）；

5、分布式测试中，通过远程启动代理服务器，默认查看结果树中的响应数据为空，只有错误信息会被报回；

6、如果并发较高，建议将Contorller机设置为只启动测试脚本和收集汇总测试结果，在配置文件里去掉Contorller机的IP；

7、分布式测试中，如果1S启动100个模拟请求，有5个Agent机，那么需要将脚本的线程数设置为20，否则模拟请求数会变成500，和预期结果相差太大。

8、jmeter-server启动报错问题，修改jmeter.properties中server.rmi.ssl.disable=true

非GUI模式运行

为什么使用非GUI模式运行jmeter

使用GUI方式启动jmeter，运行线程较多的测试时，会造成内存和CPU的大量消耗，导致客户机卡死。所以正确的打开方式是在GUI模式下调整测试脚本，再用命令行模式执行。命令行方式支持在多个环境下使用，windosw的dos环境下，也可以在linux环境上执行。

怎样使用非GUI模式

**执行路径：**执行命令前要检查当前目录是否是%JMeter\_Home%\bin 目录；如果 JMeter 脚本不在当前目录，需要指定完整的路径；如果要把执行的结果保存在其他地方也要指定完整的路径。命令中不指定测试计划与测试结果的路径时，默认都是在该目录下。

**命令：**jmeter -n -t <testplan filename> -l <listener filename>



示例

例1：测试计划与结果，都在%JMeter\_Home%\bin 目录

> jmeter -n -t test1.jmx -l result.jtl

例2：指定日志路径的：

> jmeter -n -t test1.jmx -l report\01-result.csv -j report\01-log.log

例3：默认分布式执行：

> jmeter -n -t test1.jmx -r -l report\01-result.csv -j report\01-log.log

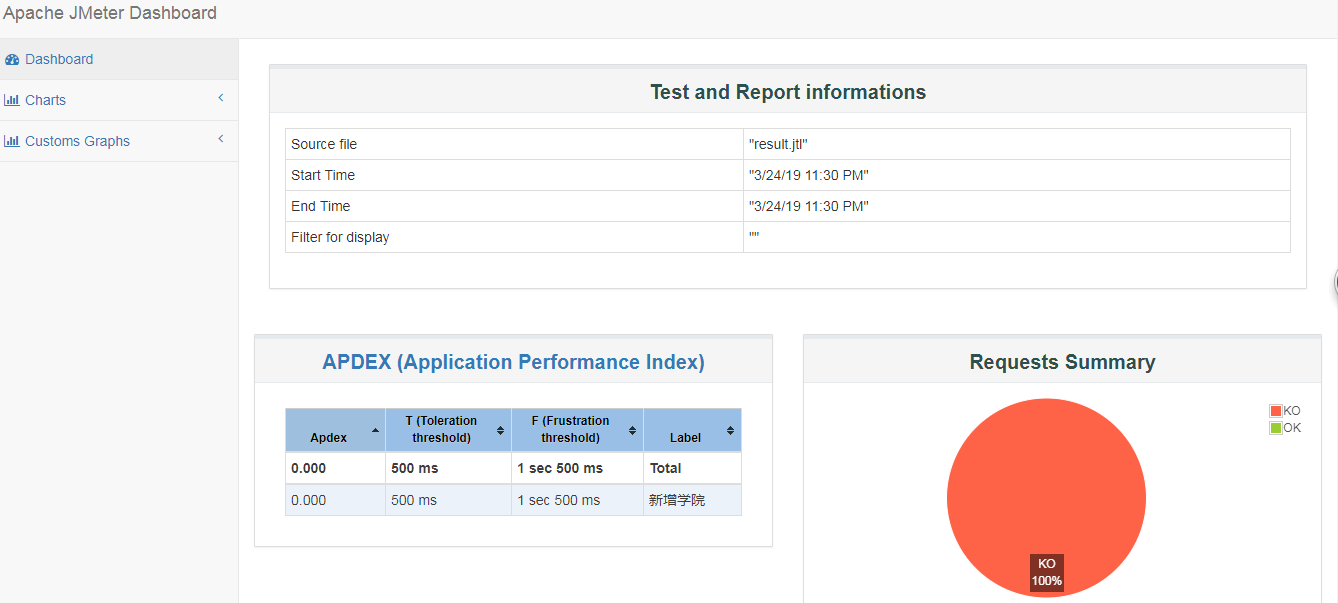
例4：指定IP分布式执行：

> jmeter -n -t test1.jmx -R 192.168.10.25:1036 -l report\01-result.csv -j report\01-log.log

例5：生成测试报表

> jmeter -n -t 【Jmx脚本位置】-l 【中间文件result.jtl位置】-e -o 【报告指定文件夹】

> jmeter -n -t test1.jmx  -l  report\01-result.jtl  -e -o tableresult



课后练习

使用jmeter请求ECShop移动端登录接口

使用jmeter完成ECShop移动端登录接口参数化

使用jmeter完成ECShop移动端购物流程

面试题

jmeter分布式如何做

jmeter如何连接数据库（mysql）

每日一练

接口测试流程

fiddler移动端抓包

扩展知识或课外阅读推荐（可选）

扩展知识

(根据时间来选择是否讲解)

课外阅读