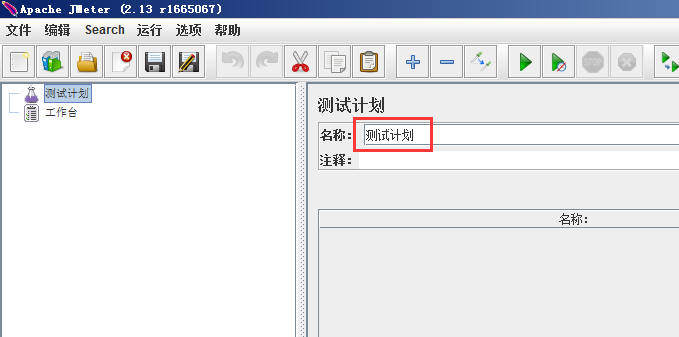
# Jmeter配置元件

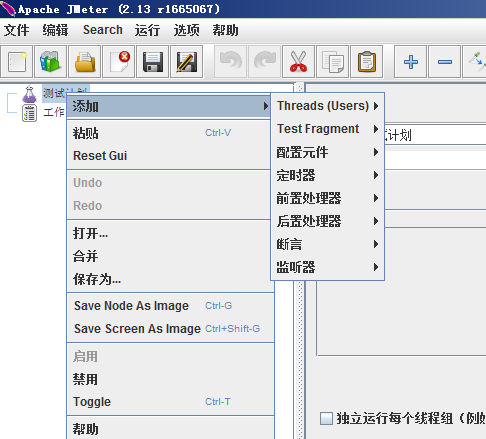
## 测试计划

打开Jmeter后，会看到jmeter默认会创建一个空的测试计划；

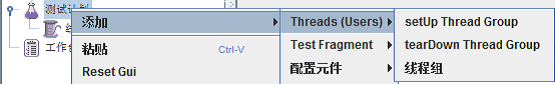


可以依据不同的项目，修改测试计划的名称，为测试设定特定的名称；

测试计划是使用 JMeter 进行测试的起点，一个测试的所有内容是于基于一个计划的。它是其它 JMeter 测试元件的容器；可以在测试计划下添加其他测试元件；



## Threads(users)



虽然有三个添加线程组的选项，名字不一样， 创建之后，其界面是完全一样的。之前的版本只有一个线程组的名字。现在多一个setUp theread Group 与terDown Thread Group。

1、setup thread group

一种特殊类型的ThreadGroup的，可用于执行预测试操作。

2、 teardown thread group

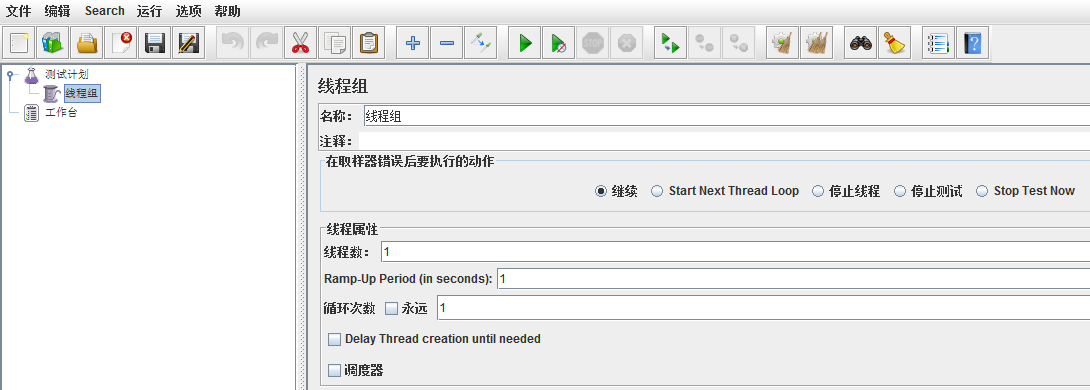
一种特殊类型的ThreadGroup的，可用于执行测试后动作。

setup thread group和teardown thread group类似于junit中的setup 和teardown，类似于loadrunner中除了action脚本之外的初始化脚本和测试完毕后对应的清除信息的脚本块。

3、 thread group(线程组)

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。

线程组是任何测试计划的起点，所有的逻辑控制器和采样器都必须放在线程组之下。其他的测试元件（如监听器）可以被直接放在测试计划之下，这些测试元件 对所有线程组都生效。线程代表一定数量的并发用户，它可以用来模拟并发用户发送请求。



用户通过线程组的控制面板可以：

* 设置线程数量
* 设置线程启动周期（Ramp-Up Period）
* 设置执行测试脚本的循环次数

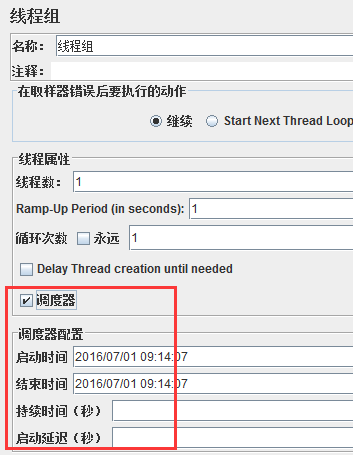
每一个JMeter线程都会完整地执行测试计划，而且它们之间是完全独立运行的。这种多线程机制被用来模拟服务器应用的并发连接。参数Ramp-Up Period告诉JMeter达到最大线程数需要多长时间。假定共有10个线程，Ramp-Up Period为100秒，那么JMeter就会在100秒内启动所有10个线程，并让它们运转起来。每一个测试线程都会在上一个线程启动10秒之后才开始运行。假定共有30个线程，Ramp-Up Period为120秒，那么线程启动的间隔就为4秒。

Ramp-Up参数不能设定得太短，否则在测试的初始阶段会给予服务器过大的压力。Ramp-Up参数也不能设定得太长，否则就会发生第一个线程已经执行完毕，而最后一个线程还没有启动的情况（除非测试人员期望这种特殊情况发生）。

如何找到一个合适的Ramp-Up参数值？建议初始值可以设定为Ramp-Up=总线程数，后续再根据实际情况适当增减。

默认情况下，JMeter线程组被设定成只执行一遍，用户可以根据实际需要设定参数"循环次数"

JMeter在1.9版本后引入了调度器，用户可以选中"调度器"选项，以便展开额外的调度器控制面板，如图所示。



在调度器控制面板中，可以设定测试运行的"启动时间"和"结束时间"。测试启动后会一直等待，直到用户设定的启动时间。测试运行期间，JMeter会在每一次循环结束后，检查是否已经达到结束时间。如果已经达到了结束时间，JMeter就会终止测试运行，否则JMeter会继续下一个测试循环。

另外，用户还可以设定"持续时间"和"启动延迟"两项参数。需要注意的是，若设置了"启动延迟"，则该设置会使"启动时间"无效，而"持续时间"会使"结束时间"无效。

## 控制器

JMeter有两种类型的控制器：取样器（sample）和逻辑控制器（Logic Controller），二者结合起来驱动了测试进程；

采样器被JMeter用来向服务器发送请求。例如，当测试人员想往服务器发送一个HTTP请求时，就加入一个HTTP请求采样器。测试人员还可以通过为采样器添加配置元件来定制化请求。

用户可以使用逻辑控制器来控制JMeter的测试逻辑，比如何时发送请求。举一个例子：测试人员可以插入交替控制器来轮流发送多个请求。

## 采样器（Sampler）



取样器（Sample）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，JMeter 原生支持多种不同的sampler ，如 HTTP Request Sampler 、 FTP  Request Sample 、TCP  Request Sample 、 JDBC Request Sampler 等，每一种不同类型的 sampler 可以根据设置的参数向服务器发出不同类型的请求。

采样器告诉JMeter发送一个请求到指定服务器，并等待服务器的请求。采样器会按照其在测试树中的顺序去执行，还可以用逻辑控制器来改变采样器运行的重复次数。

## 逻辑控制器

逻辑控制器可以帮助用户控制JMeter的测试逻辑，特别是何时发送请求。逻辑控制器可以改变其子测试元件的请求执行顺序。

## 监听器

监听器提供了对JMeter在测试期间收集到的信息的访问方法。

## 定时器

默认情况下，JMeter线程在发送请求之间没有间歇。建议为线程组添加某种定时器，以便设定请求之间应该间隔多长时间。如果测试人员不设定这种延迟，JMeter可能会在短时间内产生大量访问请求，导致服务器被大量请求所淹没。

定时器会让作用域内的每一个采样器都在执行前等待一个固定时长。如果测试人员为线程组添加了多个定时器，那么JMeter会将这些定时器的时长叠加起来，共同影响作用域范围内的采样器。定时器可以作为采样器或者逻辑控制器的子项，目的是只影响作用域内的采样器。

要在测试计划中的某个位置添加暂停，测试人员可以使用"Test Action"采样器。

## 断言

用户可以使用断言来检查从服务器获得的响应内容。通过断言可以测试服务器返回的响应与测试人员的期望是否相符。

## 配置元件

配置元件与采样器紧密关联。虽然配置元件并不发送请求（除了HTTP代理服务器例外），但它可以添加或者修改请求。

## 前置处理器

前置处理器会在采样器发出请求之前做一些特殊操作。如果前置处理器附着在某个采样器之下，那么它只会在该采样器运行之前执行。前置处理器通常用于在采样器发出请求前修改采样器的某些设置，或者更新某些变量的值（这些变量不在服务器响应中获取值）。

## 后置处理器

后置处理器会在采样器发出请求之后做一些特殊操作。如果后置处理器附着在某个采样器之下，那么它只会在该采样器运行之后执行。后置处理器通常被用来处理服务器的响应数据，特别是服务器响应中提取数据。

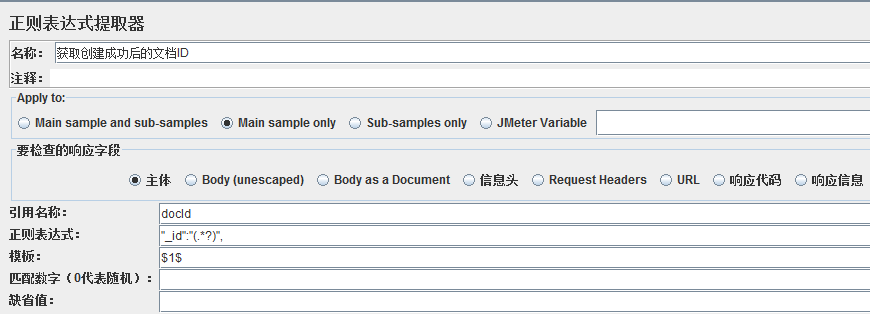
## 正则表达式

运用Jmeter正则提取器，可以从请求的响应结果中取到需要的内容，从而实现关联。Jmeter正则表达式提取器的使用方法。

1、添加Jmeter正则表达式提取器

在具体的Request下添加Jmeter正则表达式提取器（Jmeter正则表达式在“后置处理器”下面）



2、Jmeter正则表达式提取器控制面板，截图如下：

各字段解释：

名称：该正则表达式的名称，根据其意义自定义；

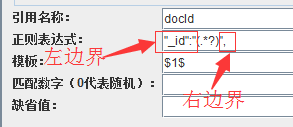
注释：对该正则表达式进行相关注释；

Apply to：默认即可；

要剪裁的响应字段：一般选择主体，即服务器返回给我们的页面主体信息；

引用名称：即参数名称，这个自己定义，在后面时需要用到该名称；

正则表达式： 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，？代表在找到第一个匹配项后停止。括号前面是所提取字段的左边界，括号后面为所提取字段的右边界；

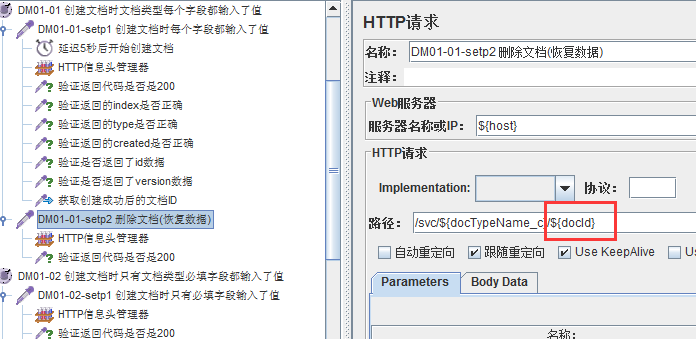


模板：$1$代表只有一组数据；

匹配数字：0代表随机取值，1代表全部取值，通常情况下填0；

缺省值：如果参数没有取得到值，那默认给一个值让它取；

在设置了正则表达式后，其调用方法如下：



# JMeter执行顺序规则

JMeter执行顺序规则如下：

配置元件

前置处理器

定时器

采样器

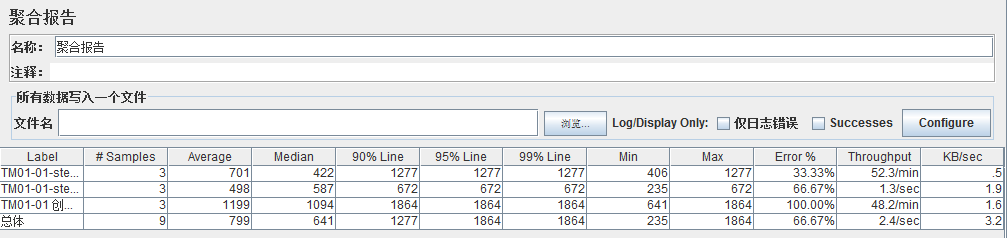
后置处理器（除非服务器响应为空）

断言（除非服务器响应为空）

监听器（除非服务器响应为空）

只有当作用域内存在采样器时，定时器、断言、前置/后置处理器才会被执行。逻辑控制器和采样器按照在测试树中出现的顺序执行。

# 聚合报告（Aggregate Report）



聚合报告是 JMeter 常用的一个监听项，其字段含义分别如下：

Label：所监控测试对应的名称；

#Samples：同一个请求取样器，发送请求的数量(注意：该值是不断累加的)。比如，请求的线程数设置为10，迭代10次，那么每运行一次测试，该值就增加10\*10=100

Average：平均响应时间——默认情况下是单个 Request 的平均响应时间

Median：中位数，也就是 50％ 用户的响应时间在该时间值内；表示响应时间不大于该时间值的请求样本数占总数的50%

90% Line：90％ 用户的响应时间在该时间值内；表示响应时间不大于该时间值的请求样本数占总数的90%

Min：最小响应时间

Max：最大响应时间

Error%：本次测试中出现错误的请求的数量/请求的总数

Throughput：吞吐量——默认情况下表示每秒完成的请求数（Request per Second）

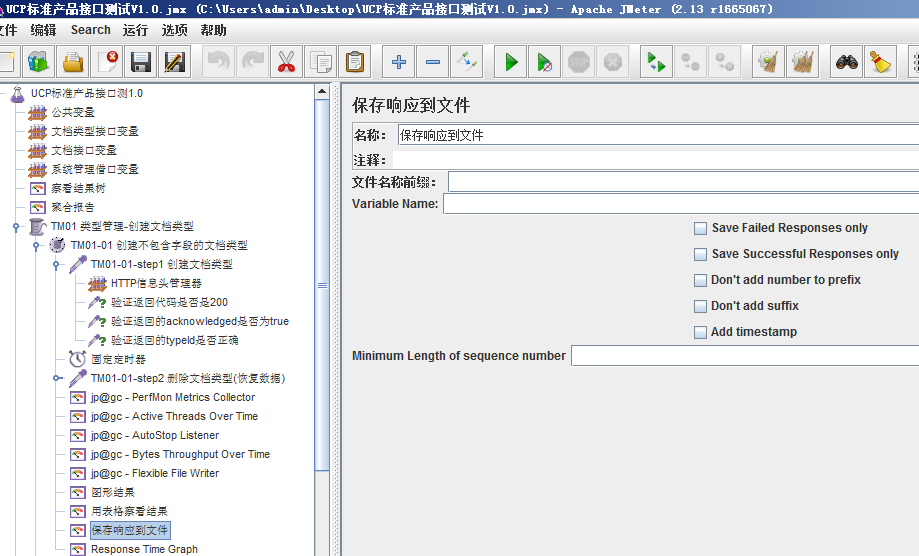
KB/Sec：每秒从服务器端接收到的数据量，相当于LoadRunner中的Throughput/Sec

注意：

1.无特别说明，以上时间的单位均为ms

2.请求响应时间指的是从client端发出请求到得到响应的整个时间

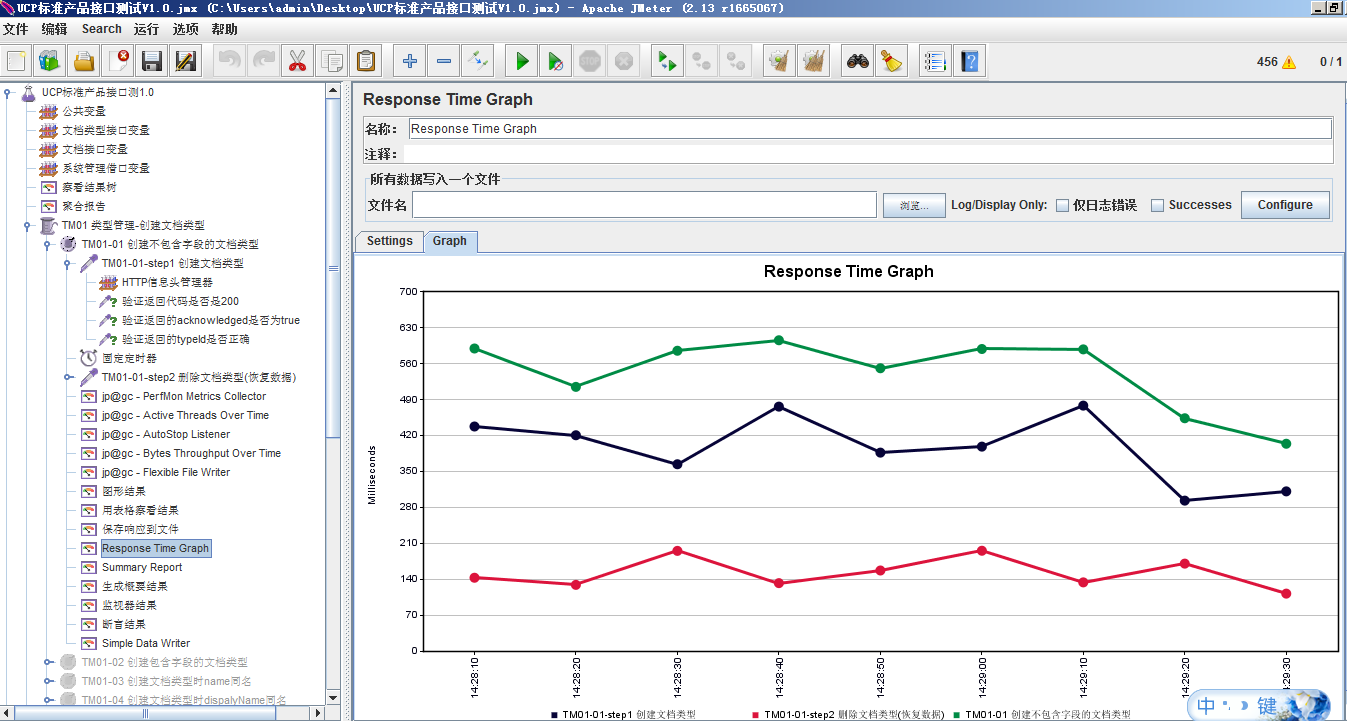
# 保存响应到文件



通过添加该监控，就会在jmeter的bin目录下生成相应的响应结果的json文件；

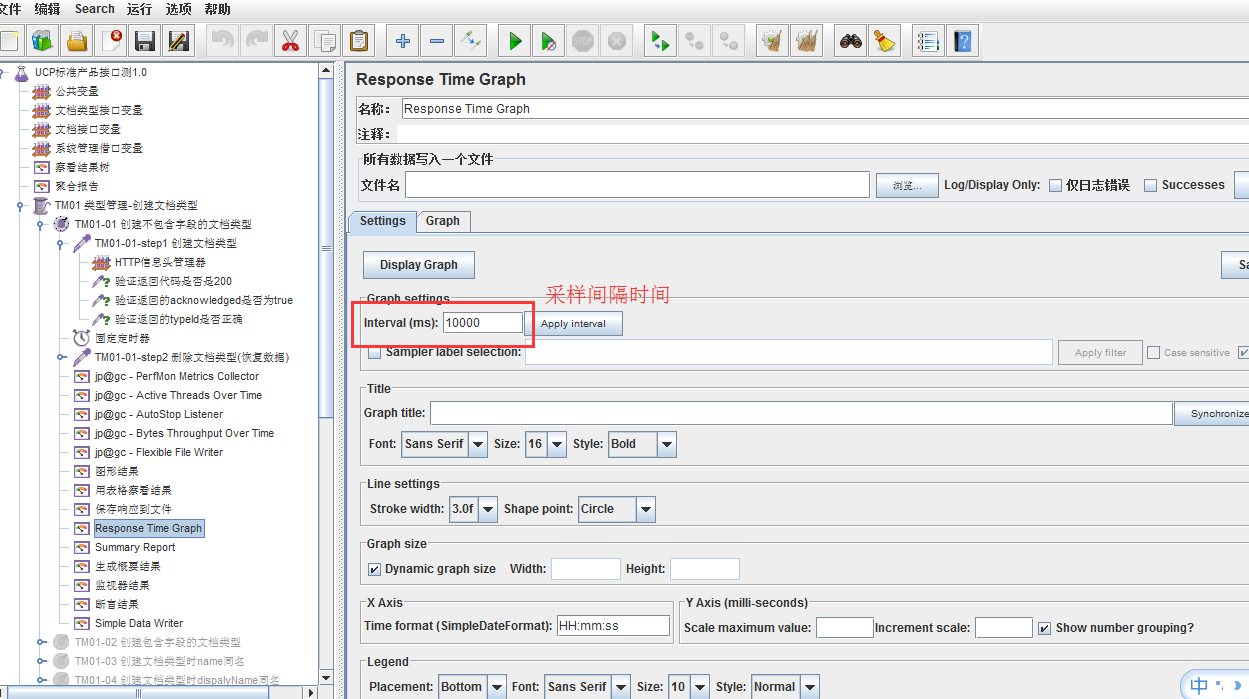
# Response Time Graph

响应时间采样图



上图展示了各操作的响应时间曲线图

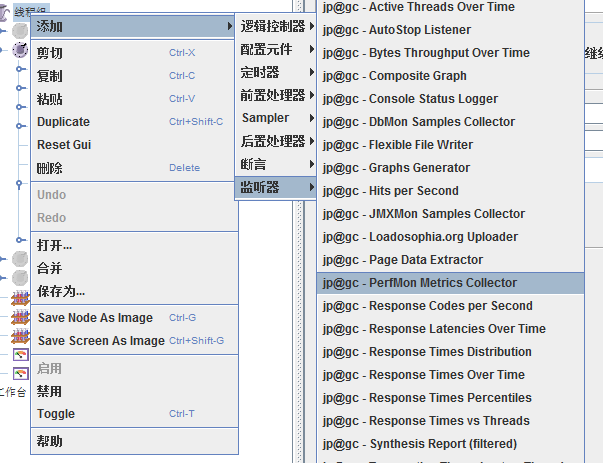
可以通过此界面进行相关设置



# JMeterPlugins

JMeterPlugins是Jmeter服务器性能监控插件，当需要监控服务器的各性能指标时，可以利用该插件来实现；插件的安装及使用步骤如下：

1. 下载插件
2. 在下载的JMeterPlugins文件夹中，将JMeterPlugins.jar包复制到Jmeter的lib目录下面的ext目录下面,重新启动Jmeter，我们点击添加就可以看到出现了很多的jp@gc-开头的文件.



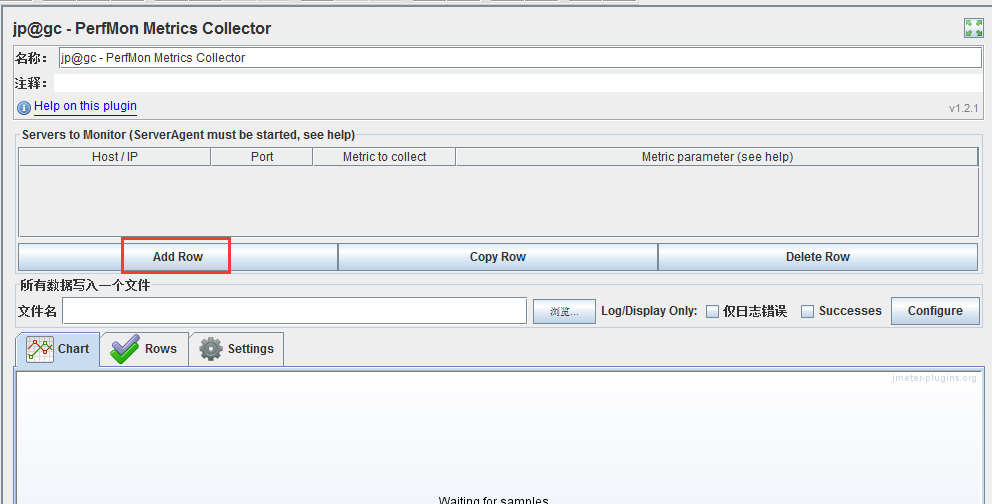
监控cpu、内存等时，使用的是jp@gc - PerfMon Metrics Collectot；

1. 在监控之前，还需要进行一步操作：将serverAgent目录及下面的文件复制到待监控的服务器上，运行/serverAgent/startAgent.bat这个文件，这样，jmeter才能成功监控服务器的性能；

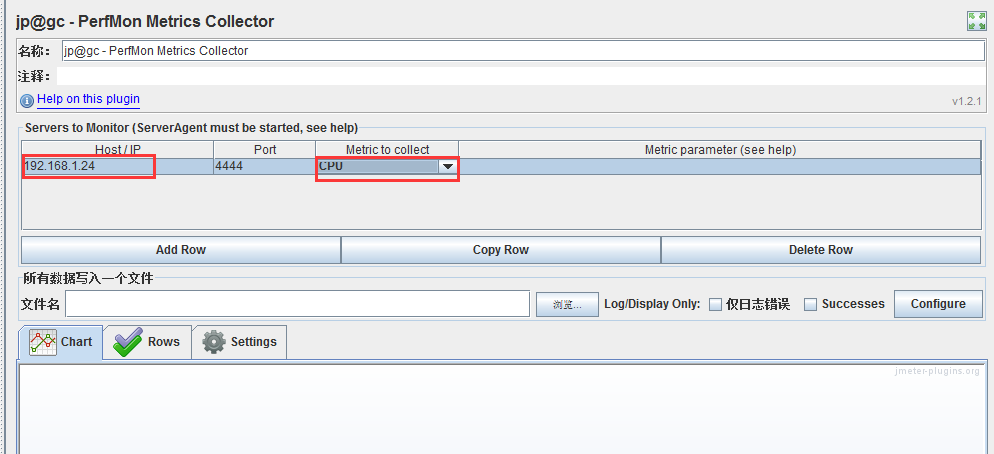
## PerfMon Metrics Collector

用于监控机器的CPU，Memory，SWAP，Disks I/O，Networks I/O

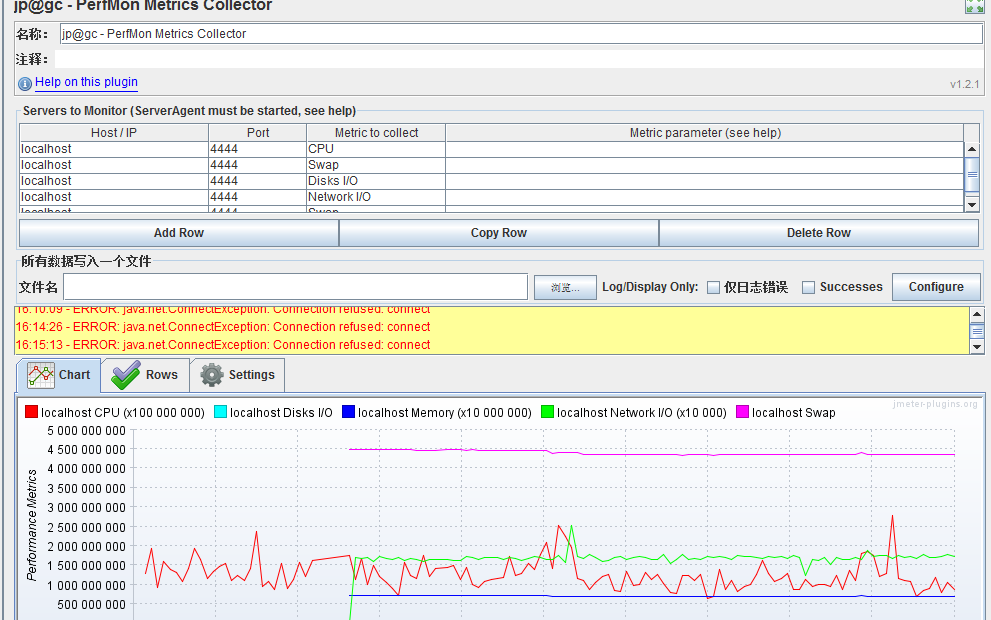
1、在jmeter中打开jp@gc - PerfMon Metrics Collectot监控器；



2、点击Add Row，添加所需要监控的服务器信息，及所需监控的服务器指标：CPU、内存等；默认端口为4444；

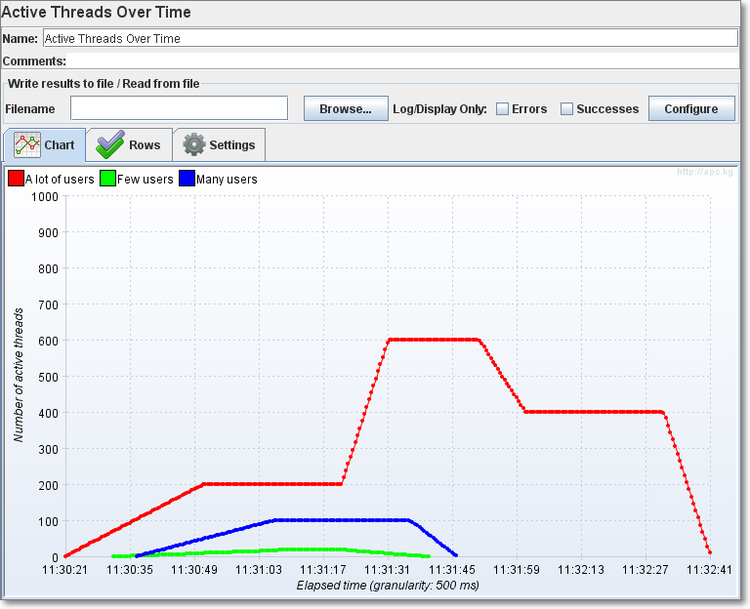


进行以上设置后，在jmeter中点击开始运行，就可以在此界面中，查看到服务器的实时性能；



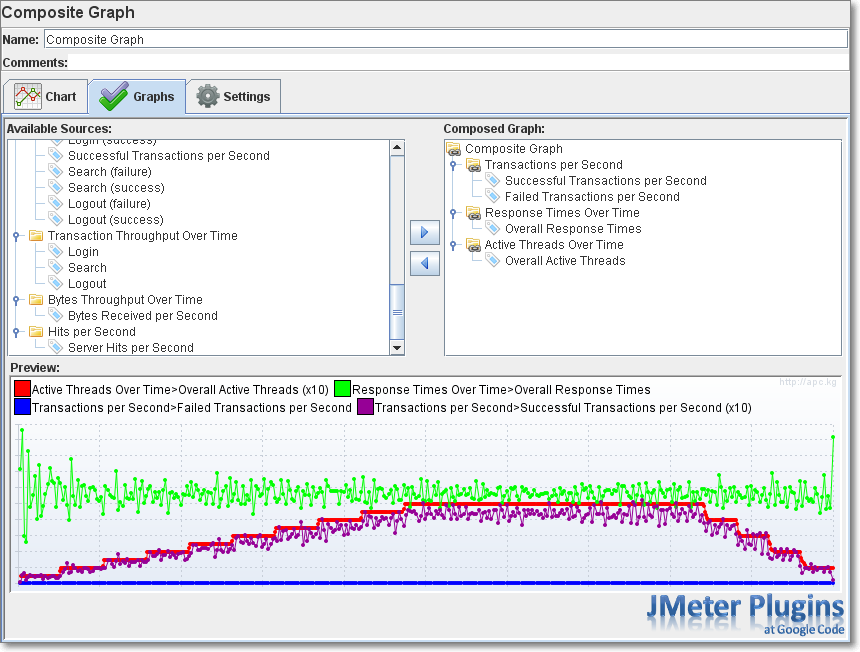
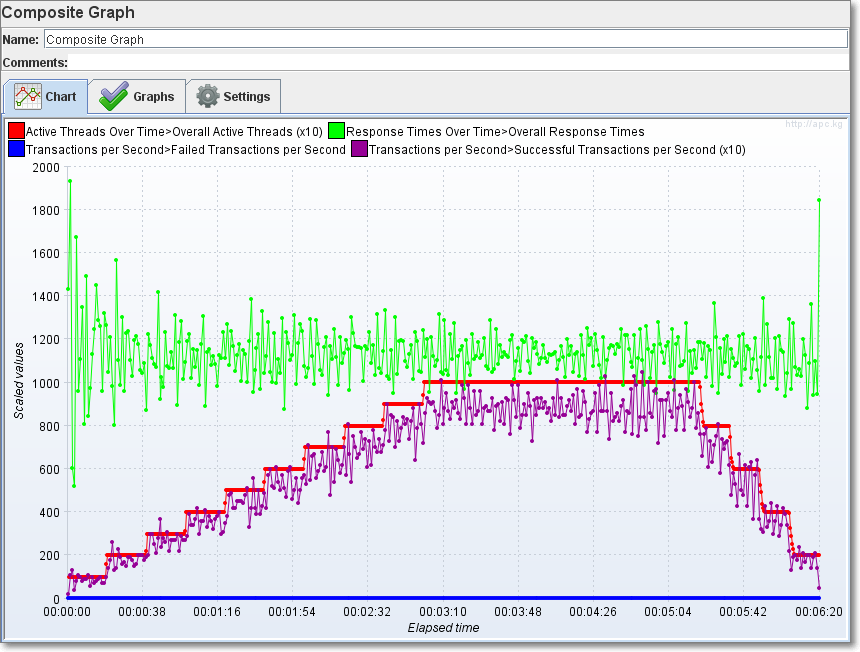
## Active Threads Over Time

每秒的活动线程数， X轴表示访问的时刻，Y轴表示活动线程数，F(X,Y)表示某个时刻的活动线程数



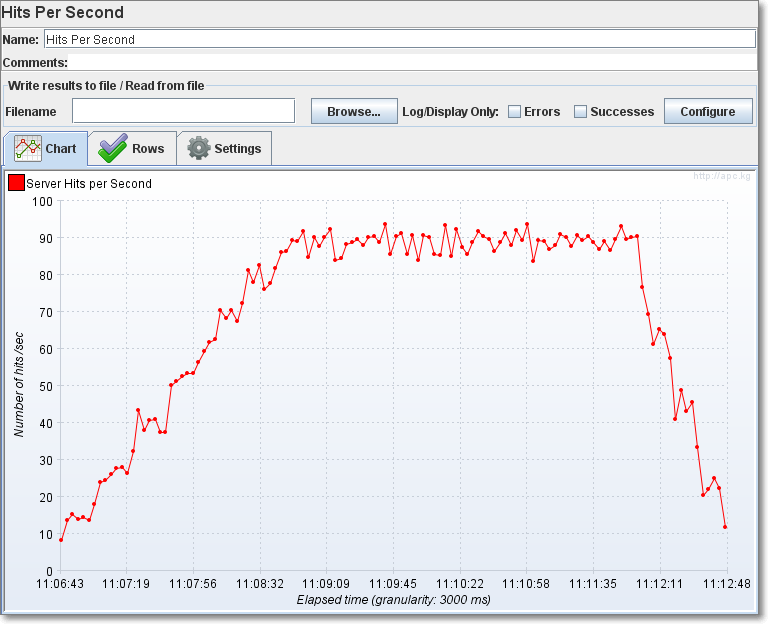
## Composite Graph

将你的测试计划中的所有的图表集合在同一张图标中以方便查看



## Hits Per Second

每秒所产生的点击服务器的次数



## Transactions per Second

每秒所完成的事务数；需与Hits per second区分；Hits per second表示每秒向服务器发送的请求数，**Transactions per Second**表示服务器针对所发送的请求，每秒成功完成的请求数；

## 

## JMeterPlugin 部分说明 - zhang_jing - zhang_jing的博客

## 

## Bytes Throughput Over Time

在压力测试期间接收和发送的bytes数

