测试报告——WordKeeper(单词管家)

厦门大学第十三届软件设计大赛决赛材料

DAOO 参赛组

组员：周兴、林联辉、陈丰铧

目录

一、引言

1.1 项目背景

单词管家是由厦门大学信息学院 2017 级学生陈丰铧、林联辉、周兴一起开发的英语学习网站。网站涉及到英语学习过程中的听、说、读、写四个方面的内容。配合深度学习的方法为用户的口语和写作进行评分，使得用户在英语学习过程中能够真正感受到自己的不足之处，从而真正提高自己的英语水平。同时网站还提供了趣味记忆单词的功能，用户可以自己指定生词集，网站为每个单词提供了有趣的图片和例句，使得复习单词成为一种乐趣。网站的单词覆盖了高中、四六级、考研到雅思、托福、SAT、GMAT、GRE 等全部英语考试词表。适合所有 12 岁以上的人群进行英语的学习。

1.2 测试目标

本次测试是针对单词管家项目进行的确认测试，目的是为了判定该系统是否满足《需求规格说明书》中规定的功能与性能指标提供的客观的依据。

二、测试概要

2.1 测试范围

序号	测试范围	测试方法	测试工具
1	单元测试	白盒/手工	IDEA
2	集成测试	手工	无
3	功能测试	自动测试	QTP
4	性能测试	自动测试	JMeter
5	验收测试	手工	无

表 1：测试范围定义表

2.2 测试环境

2.2.1 硬件环境

硬件名称	数量	性能配置	备注

联想小新笔记本电脑	1	CPU：Intel® Core™ i5-6300HQ CPU @ 2.30GHz RAM: 8.00GB	无
Sumsung S8 手机	1	CPU: 高通骁龙 835 RAM: 4GB	无
腾讯学生云服务器	1	CPU: 单核 RAM: 2GB	无

表 2：硬件环境表

2.2.2 软件环境

资源名称	版本号	备注
电脑操作系统	Windows 10	无
手机操作系统	Android OS	无
服务器操作系统	CentOS 7	无
InteliJ IDEA	2020.1.2	无
QTP	10.0	无
JMeter	5.2.1	无
Mysql	8.0.10	无

表 3：软件环境表

2.3 测试参考文档

《软件项目计划》
《用户需求说明书》
《需求规格书名书》
《概要设计说明书》

2.4 测试进度

工作阶段	开始时间	结束时间	工作量（人日）
测试计划编制	2020/04/01	2020/04/03	9
单元测试	2020/04/04	2020/04/08	12
集成测试	2020/04/09	2020/04/10	6
功能测试	2020/04/11	2020/04/14	9
性能测试	2020/04/15	2020/04/18	9
验收测试	2020/04/19	2020/04/20	6

表 4：测试进度表

三、测试设计

3.1 测试方法

单元测试采用 IDEA 软件，基于白盒测试的方法设计测试用例，以求达到 100%的代码覆盖率；
功能测试使用 QTP 自动测试化工具为项目录制脚本，采用自动化测试的好处在于之后的回归测试只需要做少量的修改就能够继续完成了，从而节省了人力成本，在功能测试中对于单词量的测试，主要与词汇大纲进行单词数量的对比；非标准考试则根据考试涉及的范围进行词汇量评估后与网站内该词汇量进行对比；记忆效果的测试参考艾宾浩斯记忆曲线对其单词记忆的分布情况进行分析；
性能测试则使用当前较为热门的 JMeter 工具通过增加网站的并发数，分析网站对于不同并发用户数量的响应情况。

3.2 测试用例设计方法

测试用例的设计基于白盒测试与黑盒测试的基本方法，其中包括：环形复杂度分析、等价类划分、边界值分析测试用例设计方法等
环形复杂度：环形复杂度是一种为程序逻辑复杂性提供定量测度的软件度量，将该度量用于计算程序的基本的独立路径数目，为确保所有语句至少执行一次的测度数量的上界。
等价类划分：等价类划分，指的是一种典型的、重要的黑盒测试方法。其就是解决如何选择适当的数据子集来代表整个数据集的问题，通过降低测试的数目去实现“合理的”覆盖，以此发现更多的软件缺陷，统计好数据后由此对软件进行改进升级。
边界值分析：边界值分析法就是对输入或输出的边界值进行测试的一种黑盒测试方法。通常边界值分析法是作为对等价类划分法的补充，这种情况下，其测试用例来自等价类的边界

3.3 测试内容

表 5：登录/登出测试

测试范围	对网站的基本登录/登出模块进行测试，看是否能够满足基本的登录提示
方法	通过对用户名和密码进行合法输入与非法输入的排列，从而测试各种情

	况下的登录反应
工具与方法	自动/手工
备注	测试中重点关注网站对于登录失败的错误提示

表 6：查询单词的测试

测试范围	测试网站能够正确给出一个单词的释义
方法	输入正确以及错误单词，查询对应的解释
工具与方法	手工
备注	网站在查询单词时会动态的给出一张与单词有关的图片

表 7：对单词量的测试

测试范围	对网站中初高中、四六级、考研、到雅思、托福、SAT、GMAT、GRE 等的项目的单词量与大纲词汇量进行比对，检验其词汇量是否一致。
方法	查阅各个英语单词项目的大纲要求，与该 APP 进行比对。
工具与方法	人工比较
备注	不同的大纲词汇可能会发生重复，注意是否排除了重复的词汇量

表 8：对记忆效果的测试

测试范围	对网站在计划内自动安排的单词记忆计划进行测试，检验其计划是否符合由短时程记忆到长时程记忆的理论依据。
方法	根据艾宾浩斯记忆曲线，人在 20 分钟、1 小时、12 小时、1 天、2 天、6 天这几个时间段会出现比较明显的记忆效果退化，所以如果在这几个时间段对需要记忆的内容进行加强，就能长期记忆某义内容。
工具与方法	对该网站在不同时间天数中出现的单词次数和时间记录下来，分析其出现的频率是否符号艾宾浩斯记忆曲线。
备注	在记忆从 STM 到 LTM 的转化中，一个重要的因素是训练间隔。实验表明，如果持续高强度的不停训练，是不能直接形成 LTM 的。唯有带上一定的间隔，才能有效将 STM 转化成 LTM。

四、单元测试

4.1 测试步骤

采用白盒测试中的“基本路径测试方法”进行单元测试：

1）设计测试用例

设计测试用例的步骤包括：画出程序流图，导出基本路径，设计具体测试用例。

2）设计测试类及运行

使用 IDEA 集成 Junit4 进行测试。

3）覆盖率测试

使用 IDEA 编译器的自带的 Run With Coverage 进行覆盖率测试，并统计类覆盖率和行代码覆盖率。

4）边界测试

设计边界测试用例进行测试，包括设计具体测试用例和进行测试。

5）行代码运行效能测试

使用 JProfiler 进行行代码运行效能测试，监控内存消耗情况、GC（JVM 垃圾回收）活动情况、类加载情况、线程情况和 CPU 消耗情况。

4.2 设计测试用例

我们以向网站中添加一本单词书为例：

4.2.1 绘制程序流程图：

```
@Test
public boolean addDictionaryTest() {
    WordBook wordBook = new WordBook();

    if(wordBook.getId() < 0)
        return false;
    else if(wordBook.getLanguage() != "english")
        return false;
    else if(wordBook.getWordNumber() <= 0)
        return false;
    else return true;
}
```

图 1：添加单词书测试代码

基于程序代码绘制出如下所示的程序流程图：

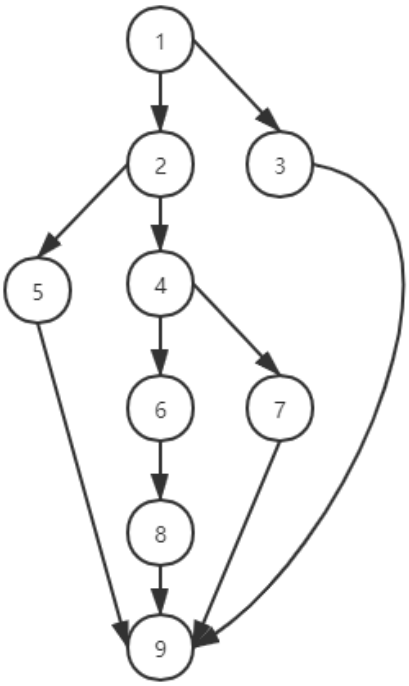


图 2：添加单词书程序流图

根据程序流图，导出基本路径如下：

环形复杂度=11 - 9 + 2 = 4

基本路径如下：

路径 1：1→2→4→6→8→9

路径 2：1→3→9

路径 3：1→2→5→9

路径 4：1→2→4→7→9

基于导出的路径我们进行测试用例的设计。

4.3 设计类运行及覆盖率测试

使用 JUnit4 编写测试类，所有测试类前加上@Transactional @Rollback 注解设置自动回滚，使得运行测试代码不会改变数据库。

```
@RunWith(SpringRunner.class)
@SpringBootTest
@AutoConfigureMockMvc
@Transactional
@Rollback
```

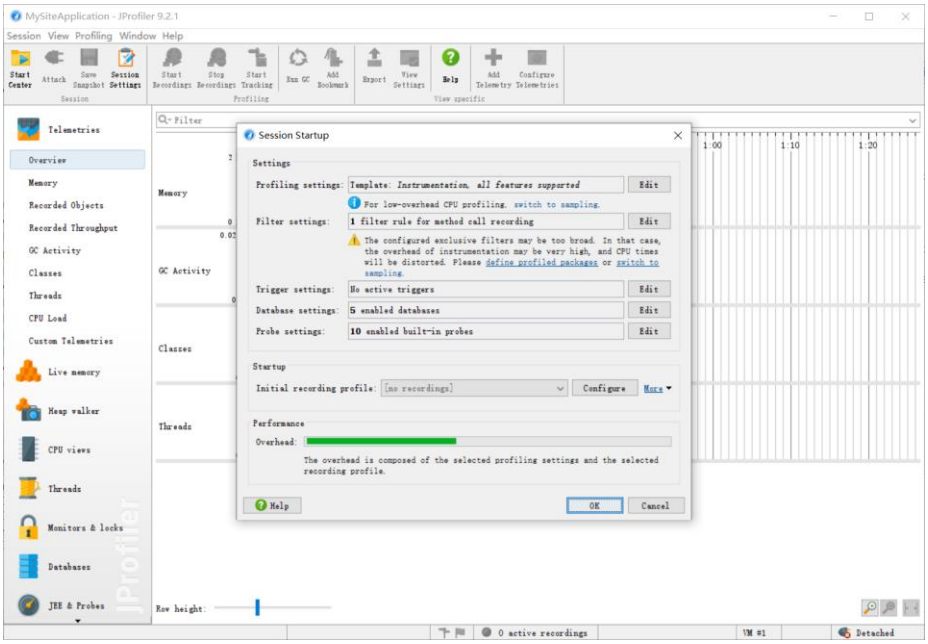
测试用例运行完成后，使用 Intellij IDEA 自带的 run coverage 进行覆盖率测试，达到代码覆盖率 100%

4.4 边界测试

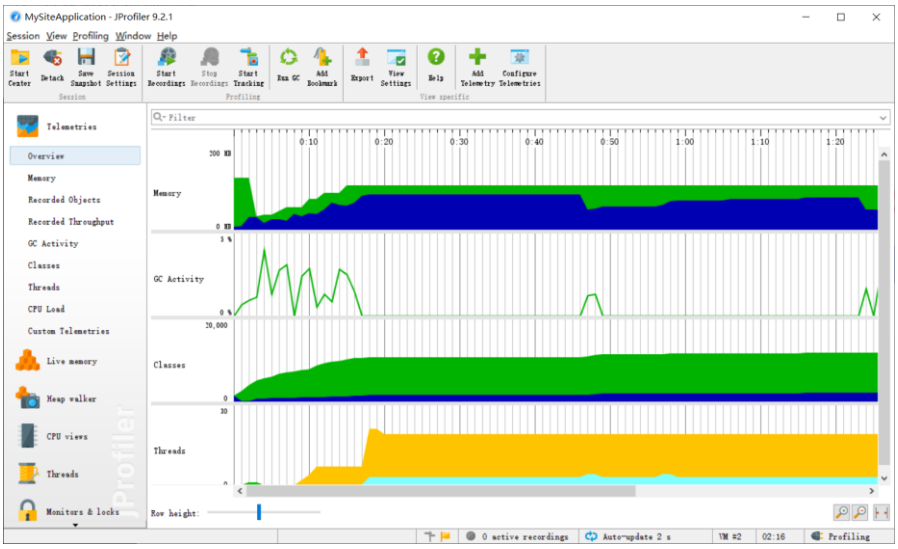
在之前的基础上对测试用例的边界值多加以测试，经验表明程序中的错误大多出现在边界值的测试用例中

4.5 行代码运行效能测试

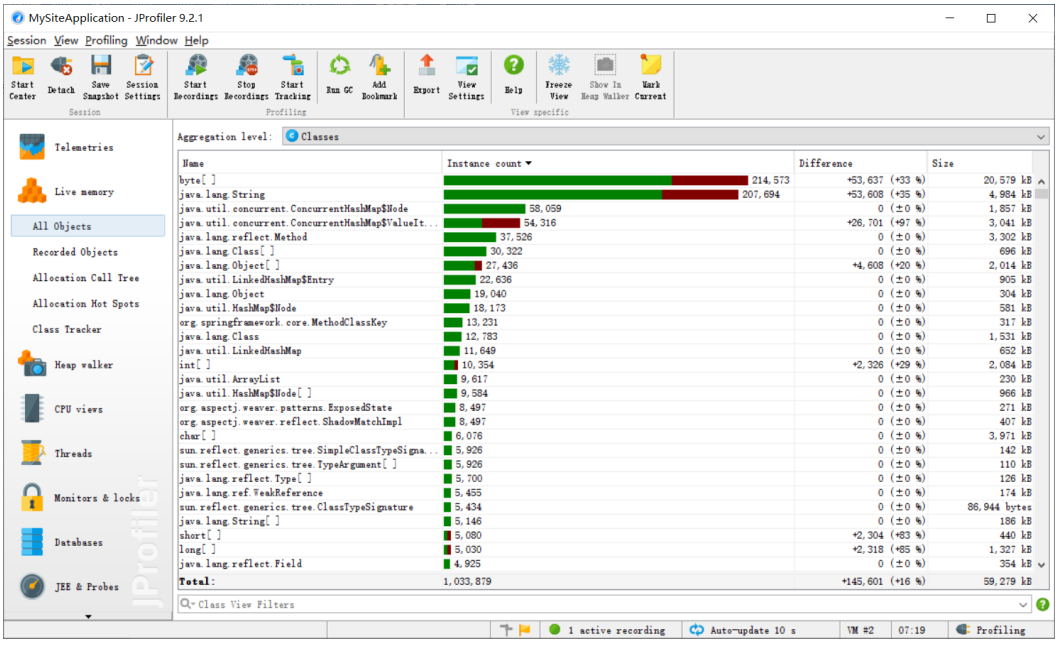
4.5.1 内存泄漏检测



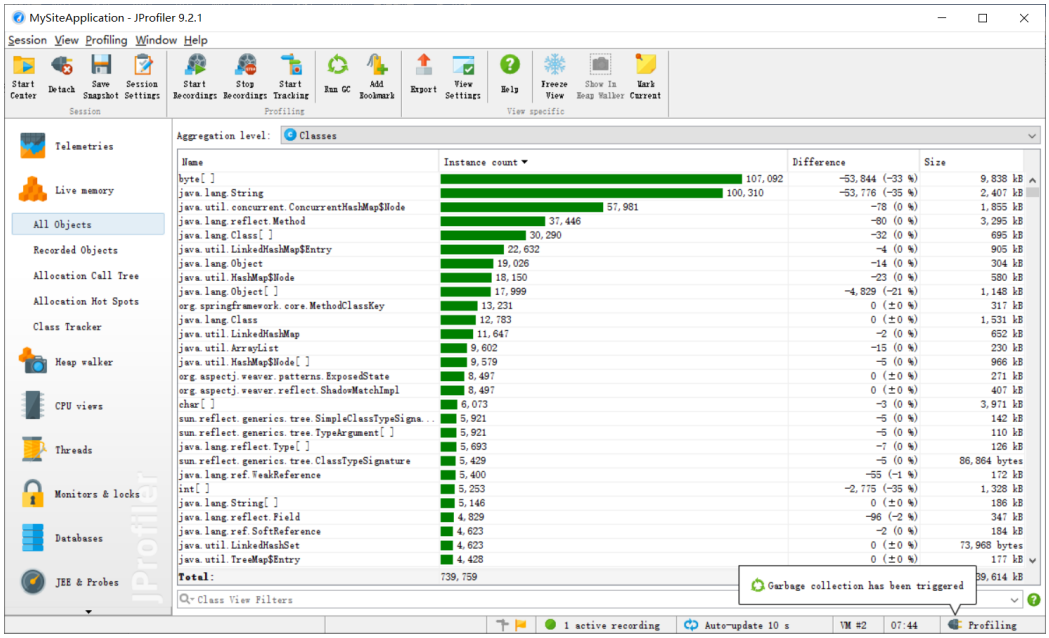
在 Java 中，分配和回收对象的内存空间在 JVM 中由垃圾回收机制自动完成。JProfiler 的 Overview 视图中，GC Activity 显示的是垃圾回收活动的时间表，如下图所示：



切换到“Live Memory-->All Objects”标签，查看当前所有对象的情况。

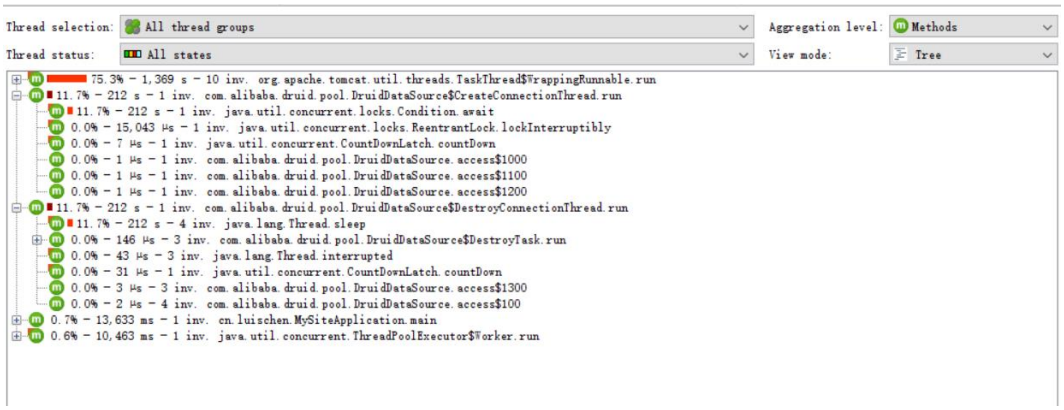


执行操作一系列可能会引起内存泄漏的操作，然后执行 GC 操作。执行垃圾回收后，仍然存在于内存中的对象有可能是泄漏的对象。一般引起泄漏的对象包括：String，char[]，HashMap 等，这些对象需要重点关注。观察后可以发现程序中大部分对象都已经被回收，并不存在内存泄漏。



4.5.2 性能瓶颈分析

执行查看新闻 100 次



分析：追踪代码操作都主要在数据库连接部分耗时，因此性能瓶颈主要在数据库部分，可以在数据库方面再进行一些优化。

4.6 单元测试总结

单元测试从代码的各个方面错程序的代码进行了一次测试，其中包括代码的逻辑、代码的效率、代码的优化等。从整体来说，程序的代码符合预期的要求，对于代码的性能方面还需进行进一步的优化，不过这目前还不是整个项目中优先级最高的部分

五、功能测试

5.1 测试步骤

1）制定测试计划

自动测试的测试计划是根据被测项目的具体需求，以及所使用的测试工具而制定的，完全用于指导测试。

2）创建测试脚本

当测试人员浏览站点或在应用程序上操作的时候，QTP 的自动录制机制能够将测试人员的每一个操作步骤及被操作的对象记录下来，自动生成测试脚本语句。

3）增强测试脚本的功能

录制脚本只是为了实现创建或者设计脚本的第一步，基本的脚本录制完毕后，

测试人员可以根据需要增加一些扩展功能，QTP 允许测试人员通过在脚本中增加或更改测试步骤来修正或自定义测试流程。

4）运行测试

QTP 从脚本的第一行开始执行语句，运行过程中会对设置的检查点进行验证，用实际数据代替参数值，并给出相应的输出结构信息。测试过程中测试人员还可以调试自己的脚本，直到脚本完全符合要求。

5）分析测试

运行结束后系统会自动生成一份详细完整的测试结果报告。

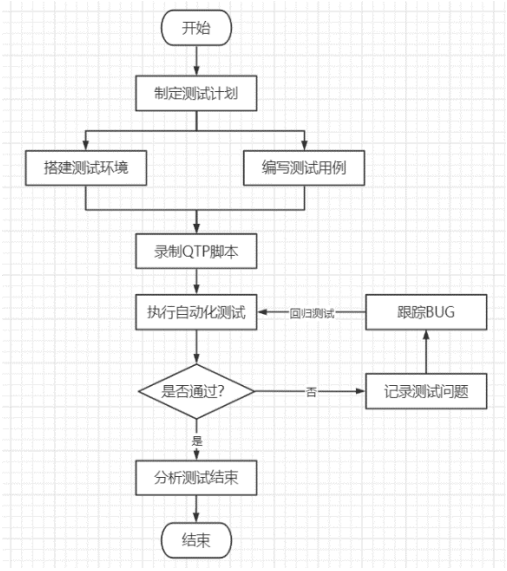


图 3：功能测试流程图

5.2 录制测试脚本

图 4：登录脚本

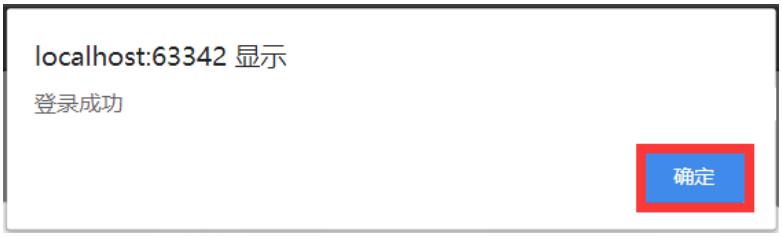


图 5：登录成功

录制完成后，得到如下所示的脚本

```
1. Browser("Word Keeper - 用户登录").Page("Word Keeper - 用户登录
   ").WebEdit("username").Set "admin"
2. Browser("Word Keeper - 用户登录").Page("Word Keeper - 用户登录
   ").WebEdit("password").SetSecure "123456"
3. Browser("Word Keeper - 用户登录").Page("Word Keeper - 用户登录").WebButton("
   登 录").Click
4. Browser("Word Keeper - 用户登录").Page("主页- Word Keeper ").Image("user-
   img").Click
5. Browser("Word Keeper - 用户登录").Page("主页- Word Keeper ").Link("注销
   ").Click
```

5.3 添加检查点

用户在输入用户密码后，点击登录按钮，如果输入正确，则会跳转到主页，如果失败，则没有任何反应。因此我们将**检查点 1** 设置在登录成功后的页面，判断登录按钮按下后的页面是否存在“注销”链接

我们使用 QTP 中的自定义检查点，在代码中设置变量，通过判断变量是否存来来判断检查点是否通过。

使用 reporter.ReportEvent 的方式来输出信息。检查点关键代码如下所示

```
1. Dim CheckLogin
2. CheckLogin = Browser("Word Keeper - 用户登录").Page("主页- Word Keeper
   ").Link("注销").Exist(3)
3. If CheckLogin = "True" Then
4.     ......
5.     reporter.ReportEvent micPass, "登录提示", "登录成功! "
6. Else
7.     reporter.ReportEvent micFail, "登录提示", "登录失败! "
8. End If
```

5.4 脚本参数化

对于脚本的参数化，我们使用 QTP 的 DataTable，首先在关键字视图下修改参数的取值方式

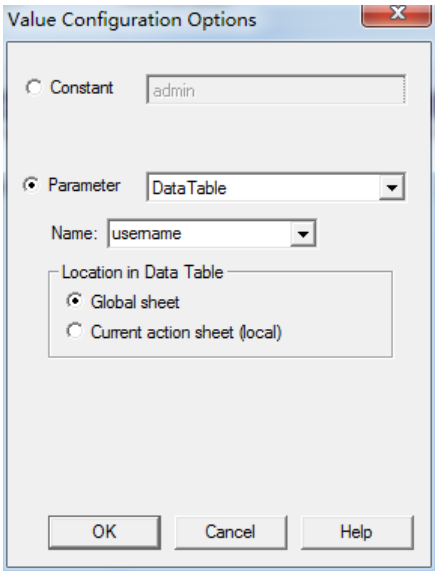


图 6: 修改参数取值方式

接着在 DataTable 中填写上我们先前设计好的测试用例

	username	password
1	admin	123456
2	admin	654321
3	adminn	123456
4		123456
5	adminnnnnn	123456
6	######	123456
7		

图 7: DataTable

修改完最终脚本的代码如下所示：

```
1. Browser("Word Keeper - 用户登录").Page("Word Keeper - 用户登录
   ").WebEdit("username").Set DataTable("username", dtGlobalSheet)
2. Browser("Word Keeper - 用户登录").Page("Word Keeper - 用户登录
   ").WebEdit("password").SetSecure DataTable("password", dtGlobalSheet)
3. Browser("Word Keeper - 用户登录").Page("Word Keeper - 用户登录").WebButton("
   登 录").Click
4. Dim CheckLogin
5. CheckLogin = Browser("Word Keeper - 用户登录").Page("主页- Word Keeper
   ").Link("注销").Exist(3)
6. If CheckLogin = "True" Then
7.     Browser("Word Keeper - 用户登录").Page("主页- Word Keeper ").Image("user-
   img").Click
```

```
8.      Browser("Word Keeper - 用户登录").Page("主页- Word Keeper ").Link("注销
    ").Click
9.      reporter.ReportEvent micPass, "登录提示", "登录成功！"
10. Else
11.     reporter.ReportEvent micFail, "登录提示", "登录失败！"
12. End If
```

5.5 测试问题及结果

问题：QTP 的自动检查点无法对对象的存在性进行检验，就算对象不存在，在运行时仅仅只会出现 Warning 而不会 Fail。
解决：通过手动编写 reporter.ReportEvent 代码来自定义检查方式，使用 Exist 函数来判断对象是否存在。

5.5.1 登陆/登出模块测试结果

测试共耗时 21 秒，被测的对象是测试用例中关于登录/登出功能的测试用例，共 7 个测试用例。从测试结果上看好像有许多测试用例都是不合格的，但仔细观察会发现这些测试用例大多是在错误提示上出现了一些问题。为此，代码上的调整并不复杂，只需简单修改几行就能将所有错误的测试用例调整过来。
记录得到的测试报告单如下：

用例编号	操作	预期结果	测试结果	测试状态
1	正确输入	进入首页	进入首页	合格
2	输入错误的密码	提示登录失败	提示登录失败	合格
3	输入不存在的用户名	提示用户名不存在	提示登录失败	不合格
4	输入空的用户名	提示输入的用户名为空	提示输入的用户名为空	合格
5	输入的用户名超出长度限制（6-11）	提示用户名长度超出限制	提示登录失败	不合格
6	输入的用户名中含有非法字	提示用户名中含有非法字符	提示登录失败	不合格

	符			
7	点击“注销”按钮	返回登录首页	返回登录首页	合格

表 9：登录登出测试结果

5.5.2 单词量模块测试结果

在标准化考试总，特别是基础类语言考试，除四六级以外，网站基本上都大于大纲所要求的词汇量。而四六级，特别是六级考试出现了词汇量相差很大的情况，可能是由于其中包括了中考出现过的单词。另外，在目前的标准化考试中，单词超纲现象比较普遍，所以网站词汇量大于大纲词汇要求是有一定的意义的。

在非标准化考试中，由于没有单词的大纲，其词汇量要求是不确定的，而网站内此类考试的词汇量也与其要求的相差较多。在一些资料中可以整理出类似与高频词汇和核心词汇这样的词汇，这些词汇数目较少，考试出现的机率较高，以这类词汇作为网站的此类考试记忆词汇是必要的。但由于词汇量较少，该网站的此类词汇覆盖面不如标准化考试广。

考试种类	大纲词汇要求	网站词汇数量	相差词汇量
中考	1680	2364	+884
高考	3500	4128	+628
四级	4200	3486	-724
六级	5300	3071	-2229
考研	5500	6287	+787
SAT	12000-15000	2950	
托福	8000	2508	-5492
雅思	7000	3272	-3728
GRE	12000-13000	2907	
GMAT	10000	3004	-6996

表 10：单词量测试结果

5.5.3 对记忆效果的测试

对于记忆效果的测试分析，不仅要结合相关的理论支持，例如艾宾浩斯记忆曲线反映的人的遗忘周期，也要考虑到图形、文字相结合的记忆方式对单词记忆的影响。同时，不同的人对于相同事物的记忆能力也不相同，要考虑到多方面的影响，最后以记忆的效果作为第一考量对数据进行分析。

5.5.4 对其他的测试

说明：以下内容均为手动测试的结果。

1）查询单词功能

用例编号	操作	预期结果	测试结果	测试状态
1	输入一个正常的单词	输出单词的多种释义	输出单词的多种释义	合格
2	输入一个错误的单词	提示找不到词语	提示找不到词语	合格
3	输入非法字符	提示找不到词语	提示找不到词语	合格

表 11：查询单词测试结果

2）修改密码功能

用例编号	操作	预期结果	测试结果	测试状态
1	合法输入	提示更改密码成功	无任何反应	不合格
2	输入合法的原密码和新密码，输入空的重复新密码	提示这是必填字段	提示这是必填字段	合格
3	输入不正确原密码	提示原密码错误	提示原密码错误	合格
4	输入的新密码与重复新密码不一致	提示你的两次输入不一致	提示你的两次输入不一致	合格
5	输入的新密码超出长度限制（6-11 位）	提示请输入长度在 6-11 位之间的字符串	提示请输入长度在 6-11 位之间的字符串	合格

表 12：修改密码测试结果

5.5 功能测试总结

本次功能测试一共涉及了 22 个测试用例的测试，测试情况如下：

被测功能	用例数	通过数	通过率
登录功能	7	4	57%
退出功能	1	1	100%
查询单词	3	3	100%
修改密码	5	4	80%

表 13：测试汇总

系统的功能测试对于大部份的功能通过率均达到 100%，而对于登录测试通过率较低的原因在于提示内容显示不当，比较容易进行修改。在测试中均与预期吻合。说明系统对于各类情况的判断考虑较为周全，且运行稳定。

被测功能的重点集中在登录功能、查询单词功能等，测试用例数较多，对于可能存在的各种组合进行了黑盒测试中的等价类划分式的测试，均为测试通过。

六、性能测试

6.1 概述

Web 性能测试是指在各种负载条件（包括正常负载、峰值负载及异常负载等）下，利用测试工具模拟运行被测系统，以此获得该系统的各种性能指标，并对所得结果加以分析的一种测试活动。随着软件系统的日益复杂，功能已不再是系统的全部，系统的性能也是系统的重要特性之一。性能测试的目的主要体现在以下几点：

（1）验证系统是否达到用户预期的性能目标。通过一种有目标的测试，以验证系统是否满足用户的要求。例如：可以通过测试，来发现某网站每一天内需要满足多大的 PV 量、多大的并发用户数、响应时间在多少秒内才能被接受等。

（2）寻找系统的最优配置，获取最小的系统成本。例如：通过不断的加压测试，可发现几核的 CPU 是最适合的，从而在可承受范围之内，对系统的成本做最小化的处理。

（3）挖掘存在的性能瓶颈，以此优化系统。例如：存在的问题是否是 SQL 执行过慢、内存泄露等原因，通过测试找到瓶颈所在，进而对系统进行优化。

6.2 性能需求

以系统登录功能为例，我们要求用户所能接受的性能指标如下所示：

并发用户数	50
平均响应时间	<=2s
百分之 90 响应时间	<=5s
吞吐量	约 20

表 14：性能指标

测试计划	线程数	Ramp-up Peried
1	10	1
2	30	1
3	50	1
4	500	1

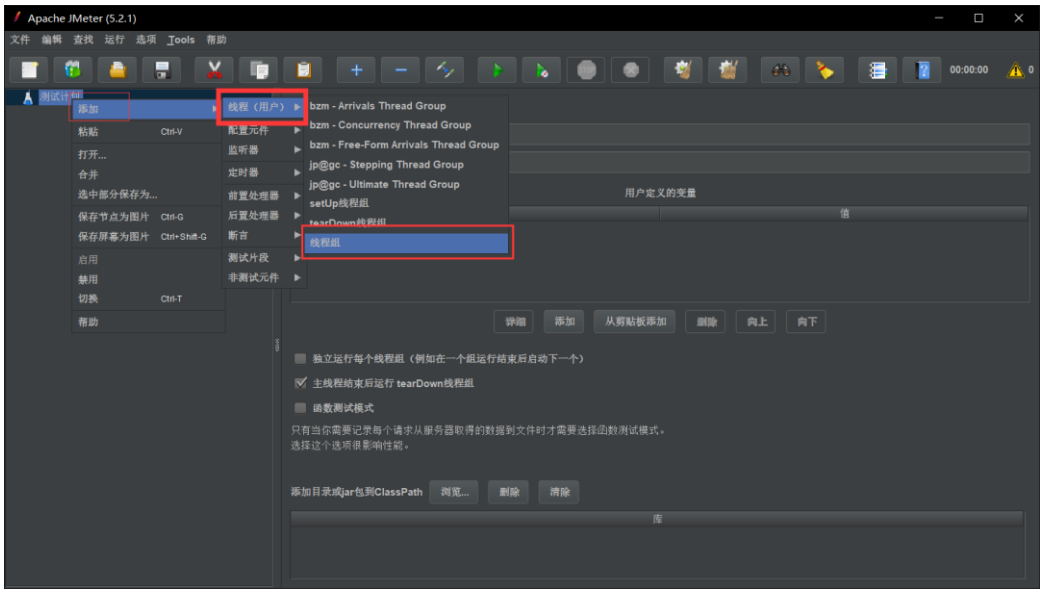
表 15：测试计划

对脚本中线程组的属性“线程数”“Ramp-Up Period（in seconds）”及“循环次数”设置不同的属性值，可体现出系统在不同负载下的性能。结合本实例系统的性能需求，可配置如表 2 所示的性能测试计划。

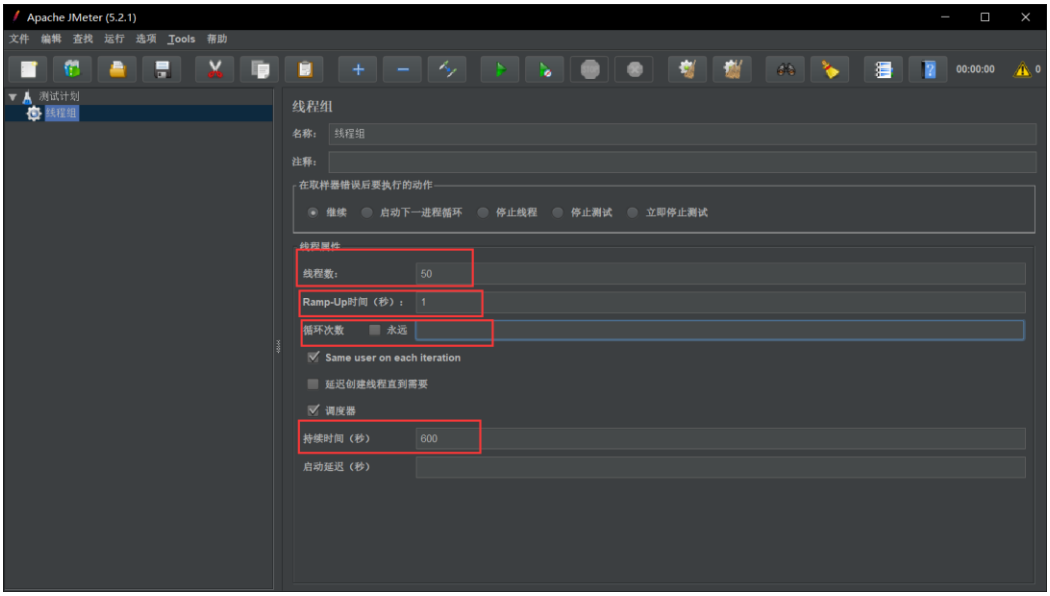
6.3 脚本录制

以系统的登录为例，录制压力测试的脚本如下所示：

1）首先点击测试计划，右键，添加线程组，如图：



线程组页面如图：



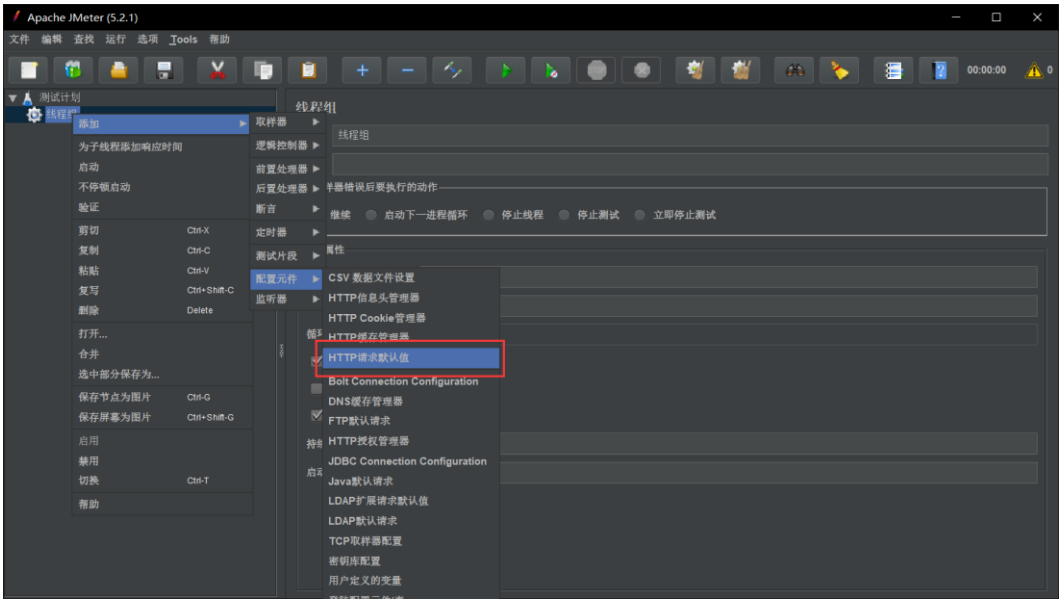
线程数：就是模仿用户并发的数量

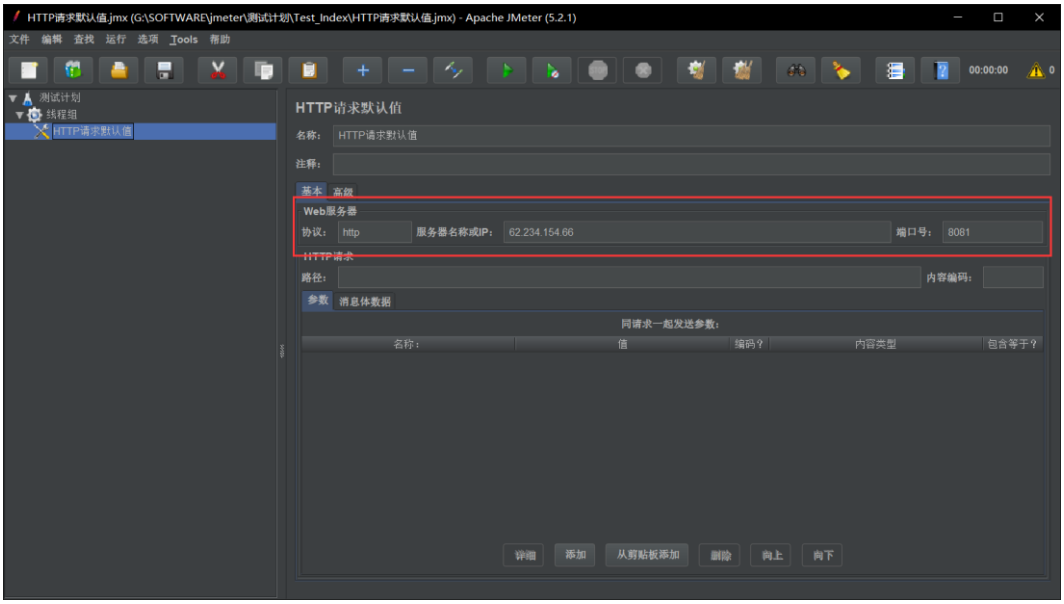
Ramp-up:运行线程的总时间，单位是秒

循环次数：就是每个线程循环多少次。

我现在的线程数是 50，就是相当于有 50 个用户，运行线程的总时间是 1 秒。也就是说在这 1 秒中之内 50 个用户同时访问，一秒钟有 50 个用户同时访问，每个用户循环一次，也就是访问一次，然后我们设置的持续时间是 600 秒，也就是 10 分钟，我们每秒有 50 用户访问首页，持续 10 分钟。

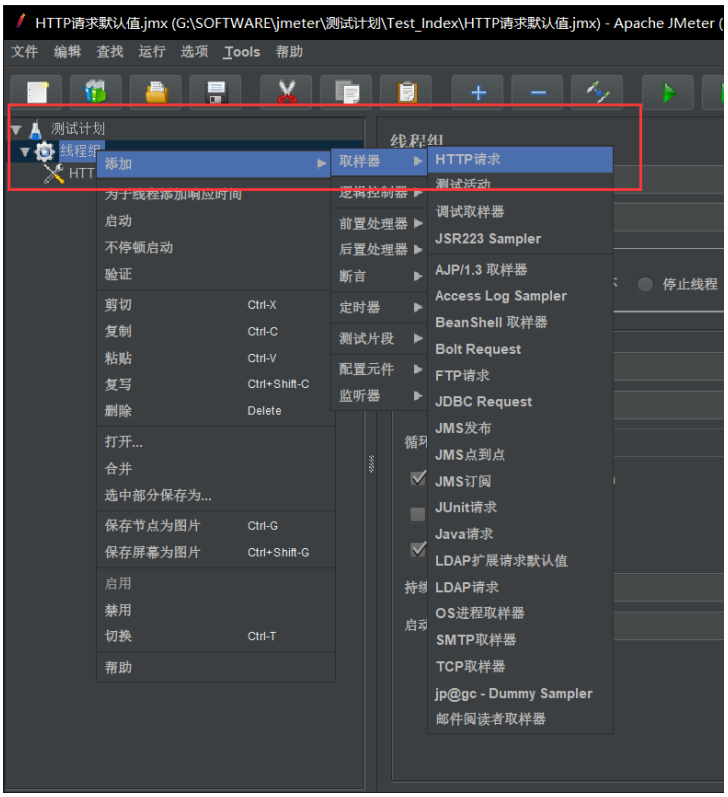
1. 点击测试计划，右键添加 HTTP 请求的默认值：如图

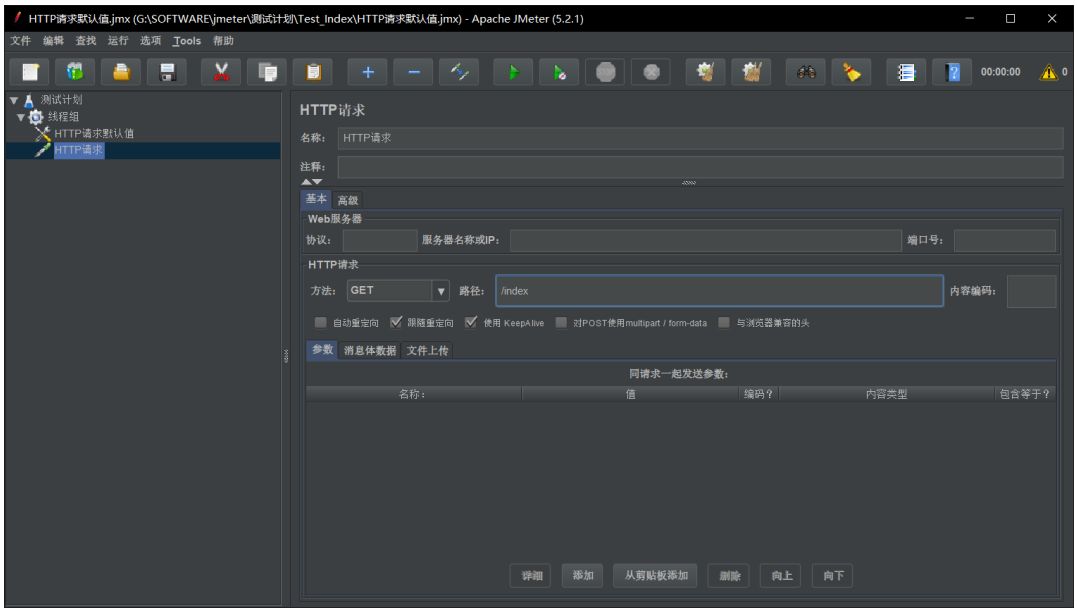




红框内填上协议 ，服务器 IP ，端口号

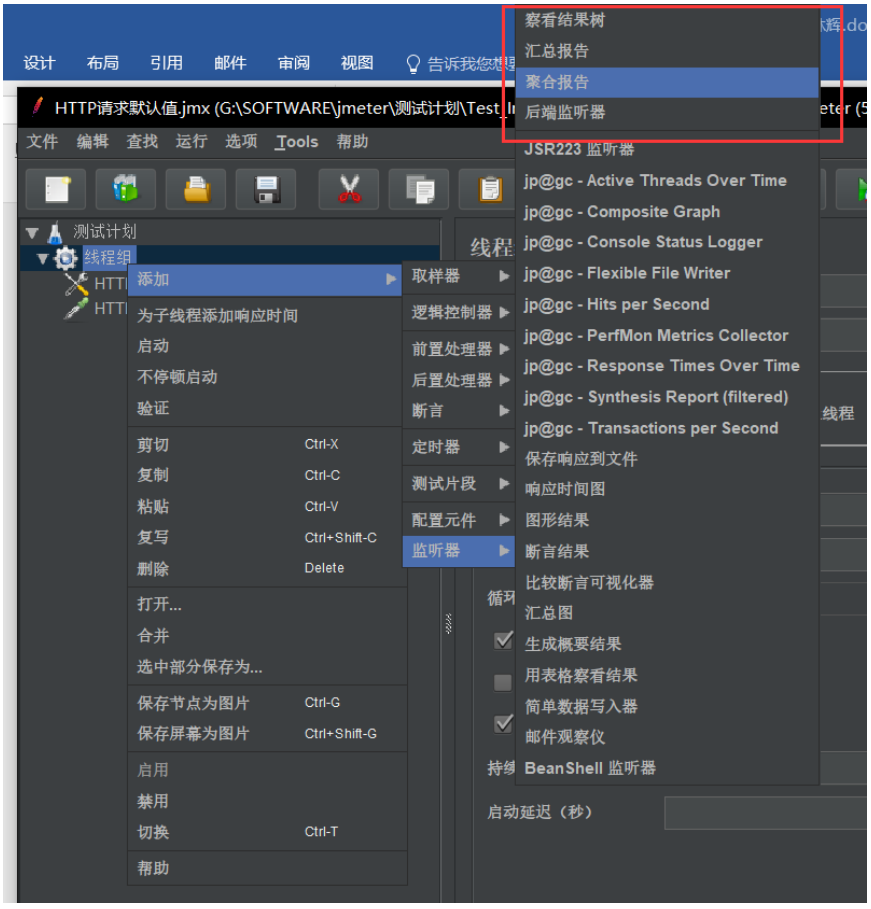
2）添加 HTTP 请求，鼠标右键点击线程组，添加 HTTP 请求：



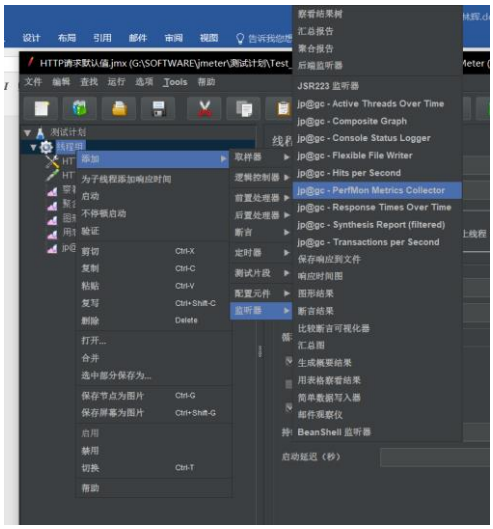


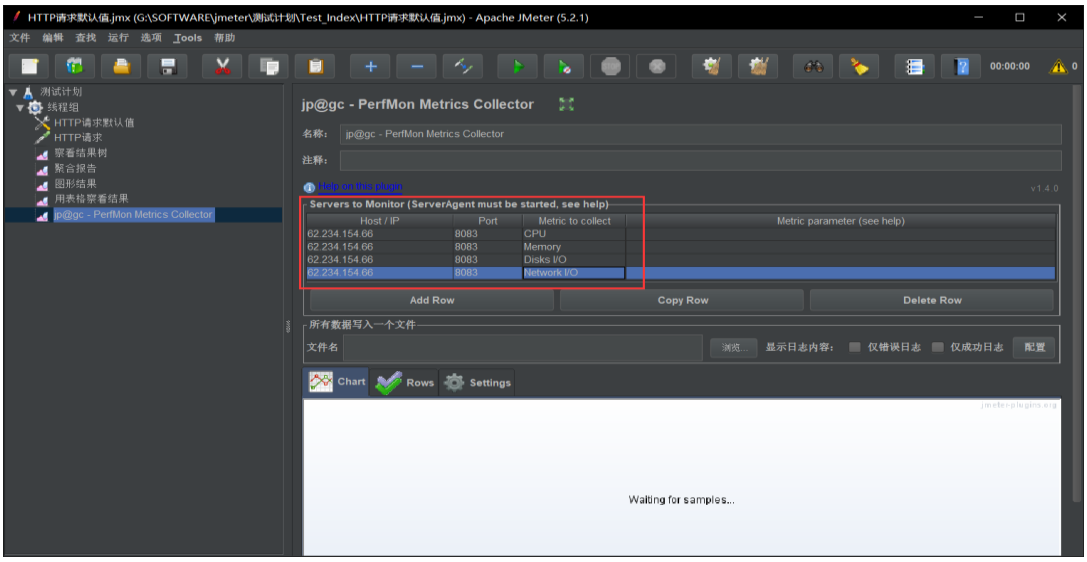
协议和 ip 地址可以不用输入，因为在 HTTP 的默认值我们已经添加了，这里只需要在路径加/表示是在根目录，不填写 IP 地址就使用的默认的。

3）添加聚合报告，查看结果树，用表格查看结果，右键点击线程，添加监听器，如图：



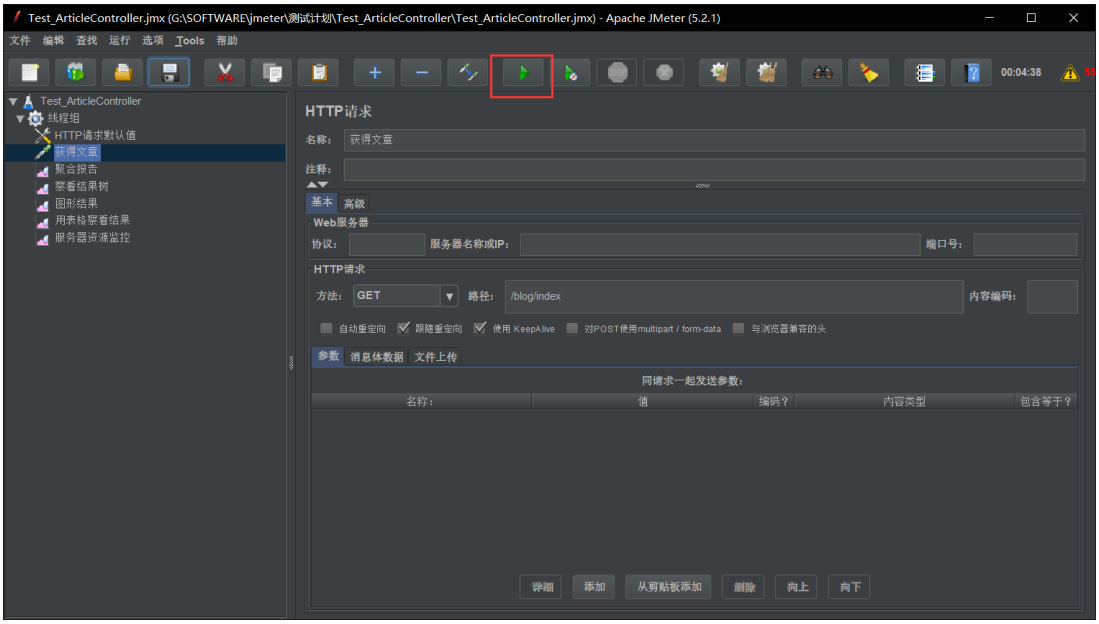
4）选择上图中的 jp@gc-PerfMon Metrics Collector，这个就是我们监控服务器资源的功能，点击之后如下图：





这个是对服务器资源的监控，可以看到，我们一共添加了四种资源监控，分别是 CPU,内存，网络 I/O，和磁盘 I/O。

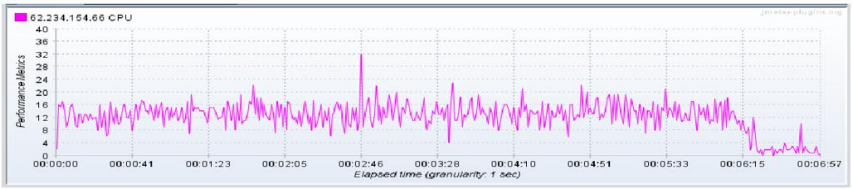
到此，我们的准备工作做完了，接下来就是按下绿色箭头进行运行脚本。



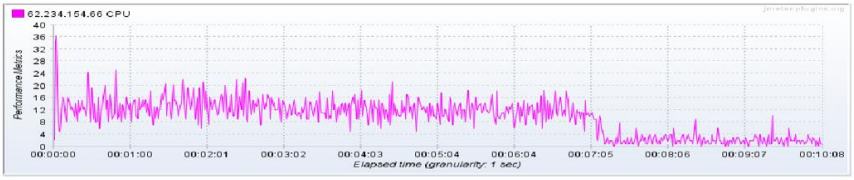
6.4测试结果及分析

1）CPU 资源

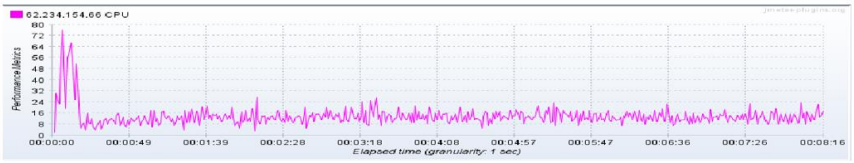
10并发量



50并发量



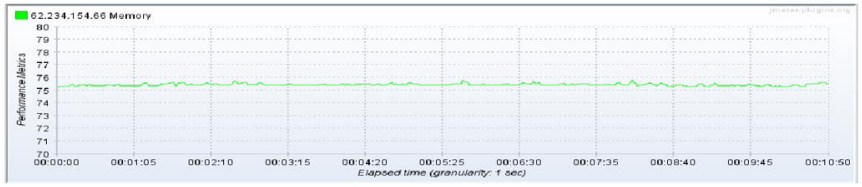
500并发量



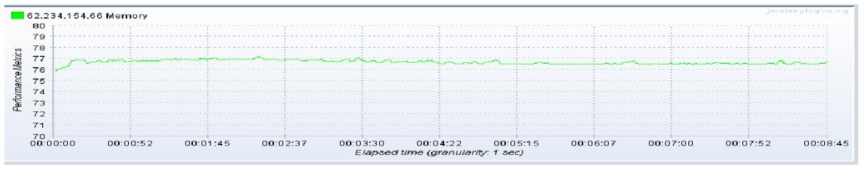
2）内存资源

10并发量

50并发量



500并发量

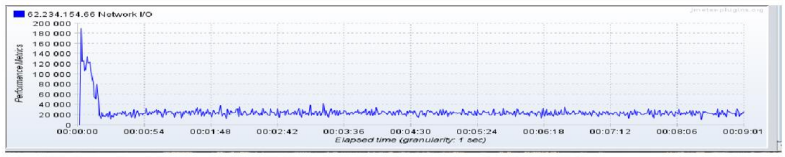


3） 网络 I/O

10并发量

50并发量

500并发量



可以看到，不管是 10,50，还是 500 并发量的时候，服务器的 CPU，内存，和网络 I/O 都是相似，提升并不明显。同样符合我们上诉的分析，也就是服务器带宽达到上限，不管并发用户量怎么提升，由于带宽受限，到达服务器的请求速率从 10 并发数时就已经达到瓶颈，这也是后续为什么 CPU 利用率，内存一直上不去的原因。所以提升服务器带宽是目前服务器最急迫的事情。

6.5 性能测试总结

通过性能测试相应的登录模块、单词查询模块、以及阅读模块，均无错误出现，虽能够保证事务正确完成，但是响应时间较长，用户体验还不够良好。影响本事务的最主要原因在于服务器网络存在瓶颈，内存占用率过高，想要投入实际使用本系统，需要提高服务器带宽，保证网络的稳定性。在网络稳定性满足的情况下，再提升性能需要提高服务器内存。

七、测试结论和建议

本次测试确认了程序中的大部分功能，程序逻辑基本无误，用户需求大体实现，之后的重点工作放在系统性能的提升，从而优化用户的使用体验

八、审批

审批意见：

测试内容比较完善，本次测试通过

签名：林联辉

签名日期：2020/07/02