

使用 Jmeter 进行性能测试

目录

目录.....	2
一、 Web 的性能测试.....	3
1. 录制脚本实现 web 性能测试.....	3
2. 执行 HTTPS 的 web 服务性能测试（暂留）.....	5
3. 直接编写 HTTP REQUEST 测试 WEB 系统及 servlet 的处理能力.....	5
4. 关于 session 问题的讨论和使用.....	6
二、 使用 JDBC 对数据库进行性能测试.....	7
三、 使用 Jmeter 的 java request 测试 java class 的性能方法.....	9
四、 在 Jmeter 的场景中使用集合点.....	13
五、 在 Jmeter 的场景中使用检查点和断言.....	14
六、 使用 Jmeter 加载变量的第一种方法：使用函数助手.....	15
七、 使用 Jmeter 加载变量的第二种方法：CSV 文件.....	16
八、 关于 JDBC 的 QUERY TYPE 的讨论.....	16

本文是属于个人技术手册一类的文档，是对 Jmeter 进行研究和之后，写下的方便记录和重新查找使用方法的文档，其中使用的参考标准为 HP 的 loadrunner，对于已经实现的方法和基本的使用情况，都有一定的介绍。本文从模块上分，可以分为：web 的录制回放执行、手动编辑 HTTP 请求、通过 JDBC 压数据库性能、通过 JAVA 请求压 class 性能等部分。具体细节上还有：jmeter 的参数化方法、集合点设置、检查点（断言）设置、关联设置、事务控制等内容。之后还要补充的有 LDAP，webservice，mail 等相关内容的描述，本文暂没涉及。

一、Web 的性能测试

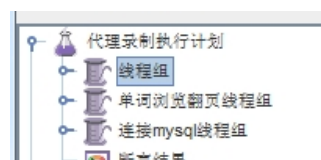
性能测试工具的最大用途，就是模拟高并发，验证 web 系统的性能，所以先从 web 的性能测试开始说起。在 jmeter 中，有两种方法可以进行 web 性能测试的准备，一个是录制出模拟脚本，然后回放进行。另一个是直接写 http 请求到 web 服务器，这个方法可以直接发参数给 servlet，可以更有针对性的验证服务器的处理能力。接下来一个一个的介绍：

1. 录制脚本实现 web 性能测试

实际上在 jmeter 中，有两种办法可以实现录制功能，一种是借助第三方软件 bad boy，通过 bad boy 录制后，导出 jmx 文件被 jmeter 使用。另一种是通过本机的代理服务，让 jmeter 捕获代理端口上 http 的各种响应，自己实现录制效果。

出于减少对第三方软件的借助，这里只介绍通过代理，让 jmeter 自己实现录制的操作。

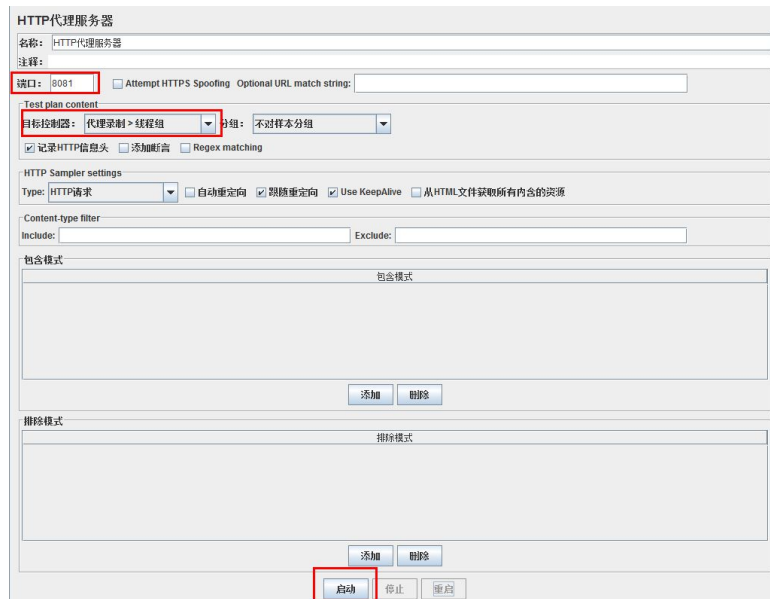
1. 在 jmeter 中创建测试计划，并添加需要的线程组：



2. 之后在“工作台”上添加一个“非测试元件”，选择“HTTP 代理服务器”

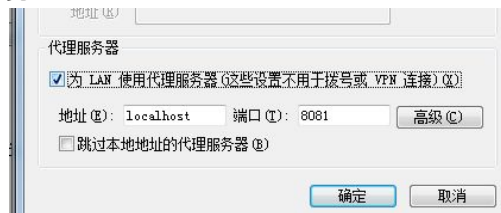


3. 对 HTTP 代理服务器进行配置



这里的端口，指的是浏览器配置的代理端口，目标控制器是指录制的内容存放在哪个线程组中。包含和排除模式，是指定录制时针对的内容，默认不选择则不生效。设置完后要启动这个功能才能生效。

4. 配置本机的代理环境



由于我就设置用我本机的 8081 端口作代理服务，故就按上图的方法设置，当然这个 IP 和地址要和 jmeter 中，http 代理服务器的填写方式一样。

5. 设置线程组中的配置元件



添加一个 HTTP 默认请求，为后面所有的 HTTP REQUEST 配置通用的属性，这里的地址和端口号，是能够通过 HTTP 协议访问到 WEB 服务的属性。

6. 开始通过代理录制脚本

因为当前本机的 HTTP 协议都会经过刚刚设置的地址和端口号，并且已经启动了 jmeter 的 HTTP 代理服务功能，因此现在所有的 HTTP 协议都会经过本机的 8081 端口并留下记录。



这里我只记录了一个页面的打开，在 HTTP 代理服务器的设置里，选择录制到哪个线程组中，访问的目录自动称为该 HTTP 请求的名称。

此时设置好线程数和执行时间，就可以进行回放了，这是最简单的通过本机代理的方式录制 web 脚本的方法。

补充：很多人可能更喜欢使用 BAD BOY 去进行录制，其实没什么区别，都是生成 jmx 文件，这里只介绍用代理去录制，目的只是想针对完全不借助第三方软件的办法，自己实现需要的功能。

但是有一个问题，因为设置了代理，所以本机上所有经过代理访问网络的 HTTP 协议，都会被记录在线程组中，这时需要根据实际情况，删除不必要的内容，同时如果可以的话，关掉当前系统上没用的网络访问也行。总之使用这个方法，最后需要自己过滤一下有用的内容。

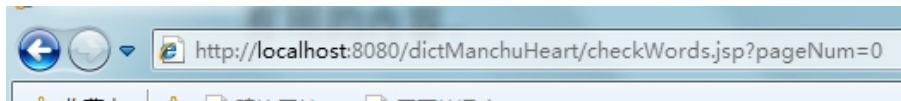
2. 执行 HTTPS 的 web 服务性能测试（暂留）

（需要研究）

3. 直接编写 HTTP REQUEST 测试 WEB 系统及 servlet 的处理能力

其实上面的部分里，我们录制的内容就是 HTTP REQUEST，这里说直接编写，其实主要针对的是 web 中需要有参数操作的两种情况：

1) 直接在地址栏里传递参数，在后面的 page 上进行解析；



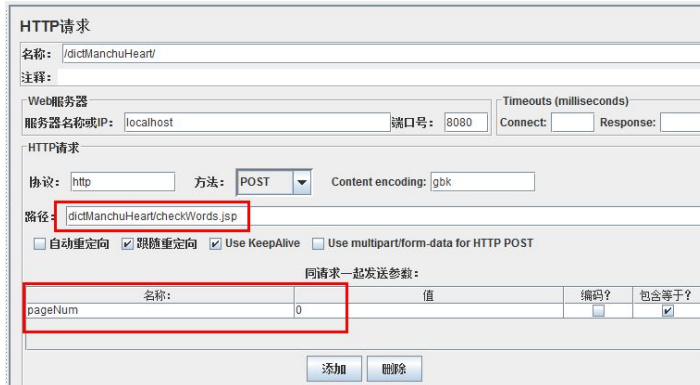
“？”后面是传递的参数名称，0 为参数，在当前页上直接进行处理，不需要经过 servlet。

2) 页面上的表单把变量提交到 servlet，处理在 servlet 中，然后跳转到指定页面；

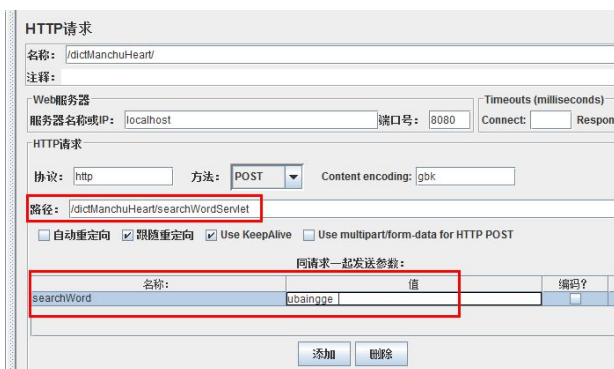


这种情况在地址栏里看不到参数，中间需要 servlet 进行处理。

其实这两种情况实现起来都一样，需要编辑线程组下的那个 HTTP REQUEST：



第一种情况：因为参数都是在地址栏里传递的，所以直接可以知道该访问哪个 jsp 文件，同时知道该传哪个参数和哪个值，当然这个值我们后面可以进行参数化。



第二种情况：需要访问的是一个 servlet 文件，这个时候要从页面源码中了解，表单提交的参数名是什么，也就是这个“searchWord”，提交的值可以参数，然后直接压这个 servlet 文件，执行效果和上面一种方法一致。

4. 关于 session 问题的讨论和使用

Session 的概念大家都知道，是 web 服务器服务器分发给每个客户端的一个会话号，在一些 web 应用里，需要使用 session 作为用户身份判断的标志，在 LR 中，session 已经是自动关联的内容了。为了更真实的模拟用户的行为和服务器的处理过程，有时候要考虑 jmeter 在 session 方面的处理。

首先，Jmeter 的线程组的概念，实际上就是虚拟用户组，他跟 session 的相同与否没有直接的联系，更多的只是把做相似功能的用户进行分类，Jmeter 线程组里的用户，及可以获得相同的 session，也可以获得不同的 session。



只有在编辑线程组的线程数的时候才能获得不同的 session，如上图，如果一个线程组安排多余 1 的线程数，那显然程序一开始执行就会得到不同的 session，但如果设置执行多次循环，却不能不断的获得新的 session，因为 session 的超时实际和 tomcat 的设置有关，根据时间设置，判断当前 session 是否超时，而是否能获得新的 sessionID。

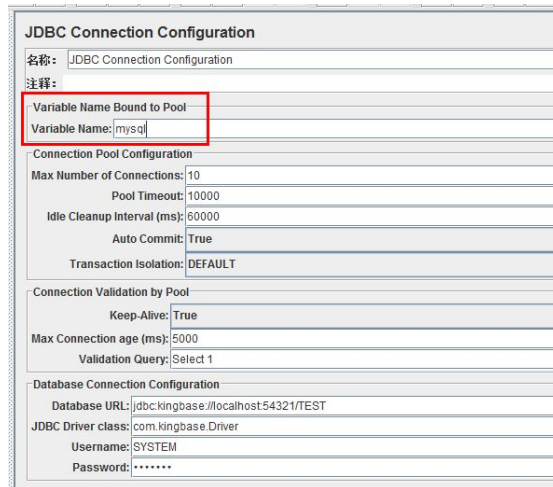


因此，在执行需要考虑 session 一致性的测试时（用户以登录的身份进行后续操作），首先要把一系列的操作放在一个线程组中，然后根据执行顺序，依靠录制或直接编写 http 请求，并且需要在第一个 http 请求之前，加上 cookie 管理器，（拖动 cookie 管理器图标到第一个 http 请求，选择添加在前。）避免回放时出现故障。

二、使用 JDBC 对数据库进行性能测试

对数据库进行性能测试，无论执行何种操作（TPCC 也好，TPCH 也好），连接方式都是类似的，要不就选择创建数据源做 ODBC 连接，要不就选择用 java 的 jdbc 连接，当然 jmeter 本身包含 jdbc 连接选项，毋庸置疑的还是选择 JDBC 的办法。

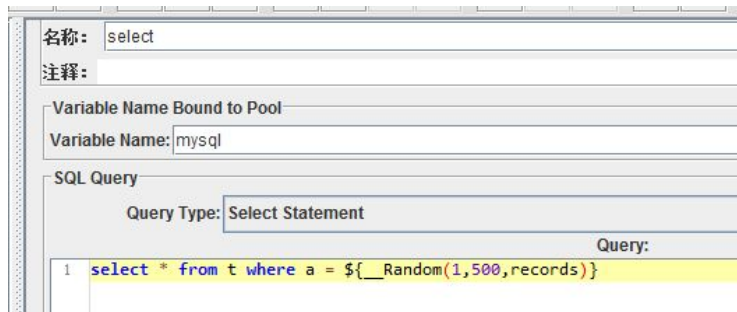
1. 使用时，首先创建一个测试计划，并为这个测试导入对应的 jdbc 连接 jar 包：



注意，要在这里给绑定的连接池定义一个名字，当然叫什么任意，但是后面的 jdbc 请求需要他。



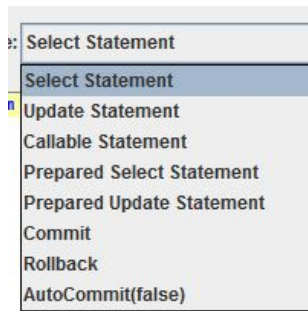
根据需要，添加 jdbc 请求，下面用一个例子做简单介绍：



截图是一个 select 类型的请求，注意绑定的连接池名称一定要填写，否则访问会报错。

这里关于参数的使用，虽然网上很多的截图表名，可以使用“？”加下面配置的办法，但我用多个版本的 jmeter 都提示莫名其妙的错误，因此我改用通过函数助手的方法生成随机数，或使用“用户参数”的办法创建需要的参数了。

再一个需要注意的是查询类型这个下拉菜单，根据 SQL 不同而不同，基本上如果涉及表修改的，就要选择 UPDATE，没有修改的就用 SELECT 的。



其中最常用的是 select statement 和 update statement。这两种查询类型中，一次只能包含一句 SQL，并且结尾没有“;”

使用 Callable statement 时，一次可以包含多个 SQL，但非结尾的 SQL 要加“;”

其余的 Prepared、commit、Rollback、AutoCommit(false)是调用 jdbc 的内部方法，配合实际 SQL 决定使用，暂没有过多研究。

三、使用 Jmeter 的 java request 测试 java class 的性能方法

对比 Loadrunner 的 java vuser，在 jmeter 中，我们可以通过使用 java request 的方法，实现对普通 java 类的测试。

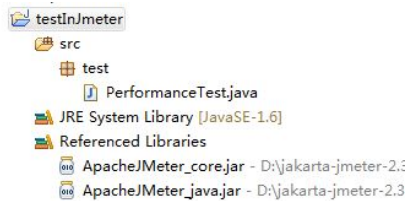
有的人说，可以利用这个功能进行白盒或灰盒测试，说实在的，白盒测试已经有 junit 的框架了，jmeter 无非就是可以直观的把需要的参数变量一次性的发送给 class，他在执行白盒测试时候的优势我倒没看出来，并且有些判断仅仅用 jmeter 的断言还不好解决，可能还须在 class 里自己写一些判断，增加了代码的复杂度，所以如果是我的话，我就老老实实用 jmeter 压性能，白盒的工作仍然交给 junit 去执行。



区别于使用 JDBC 测试数据库的性能，这里需要在线程组下添加一个 java 请求（java request），可以根据实际情况适当决定是否添加“java 请求默认值”。这是使用 jmeter 测试 class 性能的基础。

接下来是准备被测的 class 文件，和使用 junit 框架可以执行单元测试一样，使用 jmeter 的框架可以帮助我们编写测试 class 性能脚本。

首先在 eclipse 等 IDE 中建立一个 java 工程，



通过导入外部库的办法，把 jmeter 目录里，Lib 中 ext 目录中的 core 和 java 两个 jar 包导入工程。

接着在 src 下根据使用情况创建包和类，以上图为例，PerformanceTest.java 里包含要测试性能的内容。

```
1 package test;
2 import org.apache.jmeter.config.Arguments;
3 import org.apache.jmeter.protocol.java.sampler.AbstractJavaSamplerClient;
4 import org.apache.jmeter.protocol.java.sampler.JavaSamplerContext;
5 import org.apache.jmeter.samplers.SampleResult;
6
7 public class PerformanceTest extends AbstractJavaSamplerClient{
8
9     /**
10
11
12     private static long start = 0;
13     private static long end = 0;
14
15
16     * JMeter界面中可手工输入参数,代码里面通过此方法获取
17     public Arguments getDefaultParameters() {
18
19
20
21
22
23
24
25     * 执行runTest()方法前会调用此方法,可放一些初始化代码
26     public void setupTest(JavaSamplerContext arg) {
27
28
29
30
31
32
33
34
35     * JMeter测试用例入口
36     //
37     public SampleResult runTest(JavaSamplerContext arg0) {
38     public SampleResult runTest(JavaSamplerContext arg) {
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78     * 执行runTest()方法后会调用此方法.
79     public void teardownTest(JavaSamplerContext arg) {
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
260
```

```

// public SampleResult runTest(JavaSamplerContext arg) {
public SampleResult runTest(JavaSamplerContext arg) {

    SampleResult sr = new SampleResult();

    try {

        // Start
        sr.sampleStart();

        /**
         * Start~End内的代码会被JMeter
         * 纳入计算吞吐量的范围内,为了使
         * 性能结果合理,无关代码不必放此
         */

        // TODO
        int a = Integer.valueOf(arg.getParameter("a"));
        int b = Integer.valueOf(arg.getParameter("b"));
        int c = a + b;
        System.out.print("\n c = "+c);

        /**
         * True/False可按测试逻辑传值
         * JMeter会对失败次数做出统计
         */
        sr.setSuccessful(true);

        // End
        sr.sampleEnd();

    } catch (Exception e) {

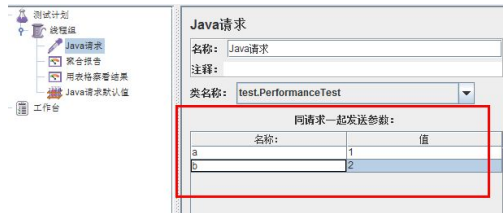
        e.printStackTrace();

    }

    return sr;
}

```

这里需要注意的是参数的传递,截图中整型变量 a 和 b 来自图形工具,使用 arg.getParameter 的方法可以取到值。



参数名称要相互对应。

```

c = 3
c = 3
c = 3cost time:0

c = 3
c = 3
c = 3cost time:0

```

执行的时候,根据 class 中的代码,可以将执行结果或特定信息打印到后台日志。

全部代码:

```

package test;

import org.apache.jmeter.config.Arguments;

import org.apache.jmeter.protocol.java.sampler.AbstractJavaSamplerClient;

import org.apache.jmeter.protocol.java.sampler.JavaSamplerContext;

import org.apache.jmeter.samplers.SampleResult;

public class PerformanceTest extends AbstractJavaSamplerClient{

    /**
     *

```

```
*/  
  
private static long start = 0;  
private static long end = 0;  
  
/**  
 * JMeter界面中可手工输入参数,代码里面通过此方法获取  
 */  
public Arguments getDefaultParameters() {  
  
    Arguments args = new Arguments();  
  
    return args;  
}  
  
/**  
 * 执行runTest()方法前会调用此方法,可放一些初始化代码  
 */  
public void setupTest(JavaSamplerContext arg) {  
  
    // 开始时间  
    start = System.currentTimeMillis();  
}  
  
/**  
 * JMeter测试用例入口  
 */  
// public SampleResult runTest(JavaSamplerContext arg0) {  
public SampleResult runTest(JavaSamplerContext arg) {  
  
    SampleResult sr = new SampleResult();  
  
    try {  
  
        // Start  
        sr.sampleStart();  
  
        /**  
         * Start~End内的代码会被JMeter  
         * 纳入计算吞吐量的范围内,为了使  
         * 性能结果合理,无关代码不必放此  
         */  
  
        // TODO  
        int a = Integer.valueOf(arg.getParameter("a"));  
        int b = Integer.valueOf(arg.getParameter("b"));
```

```

        int c = a + b;
        System.out.print("\n c = "+c);

        /**
         * True/False可按测试逻辑传值
         * JMeter会对失败次数做出统计
         */
        sr.setSuccessful(true);

        // End
        sr.sampleEnd();

    } catch (Exception e) {

        e.printStackTrace();

    }

    return sr;

}

/**
 * 执行runTest()方法后会调用此方法.
 */
public void teardownTest(JavaSamplerContext arg) {

    // 结束时间
    end = System.currentTimeMillis();

    // 总体耗时
    System.err.println("cost time:" + (end - start) / 1000);

}
}

```

四、在 Jmeter 的场景中使用集合点

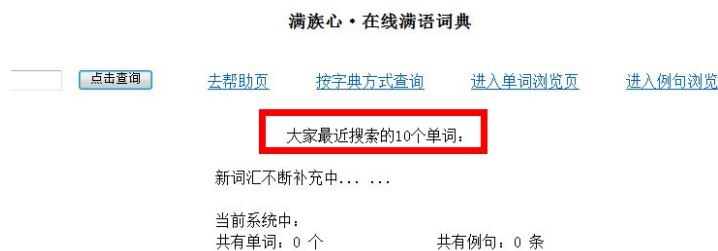
使用“固定定时器”模拟集合点效果



五、在 Jmeter 的场景中使用检查点和断言



最常用的是“响应断言”，加在要检查的响应信息下面。



要检查的页面和内容



在 Jmeter 上添加了对应的内容



断言成功：不提示有错误



断言失败：提示文本域找不到

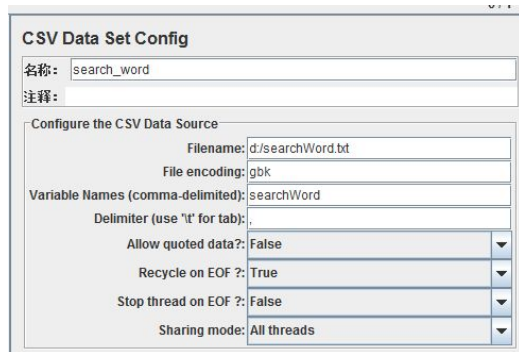
六、使用 Jmeter 加载变量的第一种方法： 使用函数助手



七、使用 Jmeter 加载变量的第二种方法： CSV 文件



```
Hosika ufuhi  
emu uksura  
emu ulqin jiha  
fi i ulgakuu  
fi ulgabumbi  
fi ulgambi  
geren de ulhiquke obuki.  
giyangkv ubiyada  
hexen ulhun  
jakvn ubu
```



八、关于 JDBC 的 QUERY TYPE 的讨论

在 jmeter 的 jdbc 请求中，可以选择不同的 Query Type，根据不同的选择，可以创建复杂的查询场景。

JDBC Request

名称: select

注释:

Variable Name Bound to Pool

Variable Name: mysql

SQL Query

Query Type: **Select Statement**

select * from t where

Select Statement

Update Statement

Callable Statement

Prepared Select Statement

Prepared Update Statement

Commit

Rollback

AutoCommit(false)

SQL Query

Query Type: **Select Statement**

select * from t where

Select Statement

Update Statement

JDBC 最基本的两个操作，查询和更新。

select: `ResultSet rs = stmt.executeQuery(sql) ;`

update : `int rows = stmt.executeUpdate("INSERT INTO ORDERS (ISBN, CUSTOMERID) VALUES(195123018, 'BILLING')", Statement.RETURN_GENERATED_KEYS);`

Callable Statement

Prepared Select Statement

Prepared Update Statement

JDBC 的两个预编译语句，执行效果和前面两个相同，但是相对更安全，避免攻击操作。

prepared select: `ResultSet rs = PreparedStatement pstmt .executeQuery(sql) ;`

prepared update : `PreparedStatement pstmt = conn.prepareStatement("UPDATE EMPLOYEES SET SALARY = ? WHERE ID = ?");`

Callable Statement

Prepared Select Statement

Prepared Update Statement

`create table test(a int);`

`insert into test values(1);`

`create table temp(b int);`

SQL Query

Query Type: **Update Statement**

create or replace procedure proc_1() as
declare
para1 int;
begin
select a into para1 from test;
insert into temp values(para1);
end;

variable name: proc_1

SQL Query

Query Type: Callable Statement

```
call proc_1();
```