

Jmeter训练营

@北河

认识码同学

码同学专注于软件测试领域提升培训。为众多一线IT企业输送了3000多位软件测试高端人才，为软件测试行业整体能力提升做出了卓越的贡献。

每年上万余计的测试同学，通过参加码同学高质量的技能培训，从而改变了自己的人生。他们中，有的人从技术小白成为技术大咖，有的人从月薪八千直接升级至月薪过两万，有的人从一线职员升级为测试经理。

携手强大的师资团队，只为成就更优秀的你





北河老师

10+测试行业经验，性能测试专家、测试开发专家

曾供职于阿里、京东等多家大型互联网公司，对互联网/电商行业的技术架构非常熟悉

多年的专职性能测试经验，超过百个性能项目的实战经验，多次参与百万级TPS的压测项目

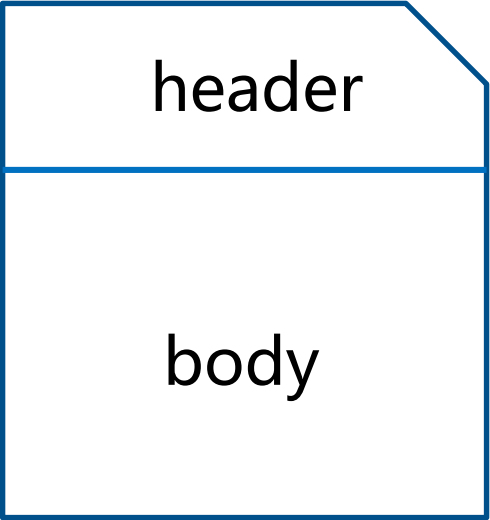
擅长复杂性能场景设计、性能瓶颈定位与调优。在公司主导设计开发了多个测试平台



- 1、支持客户/服务器模式。
- 2、简单快速：客户向服务器请求服务时，只需传送请求方法和路径。请求方法常用的有GET、POST、PUT、DELETE。每种方法规定了客户与服务器联系的类型不同。
由于HTTP协议简单，使得HTTP服务器的程序规模小，因而通信速度很快。
- 3、灵活：HTTP允许传输任意类型的数据对象。正在传输的类型由Content-Type加以标记。
- 4、无连接：无连接的含义是限制每次连接只处理一个请求。服务器处理完客户的请求，并收到客户的应答后，即断开连接。采用这种方式可以节省传输时间。
- 5、无状态：HTTP协议是无状态协议。无状态是指协议对于事务处理没有记忆能力。缺少状态意味着如果后续处理需要前面的信息，则它必须重传，这样可能导致每次连接传送的数据量增大。另一方面，在服务器不需要先前信息时它的应答就较快。



数据包结构





请求header

```
GET / HTTP/1.1
Host: www.baidu.com
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate, sdch, br
Accept-Language: zh-CN,zh;q=0.8
Cookie: BAIDUID=A844A3DF803EBAED54BF6A6897E6AB36:FG=1; BIDUPSID=A844A3DF803EBAED54BF6A6897E6AB36; PSTM=1561788493; delPer=0; BD_HOME=0; H_PS_PSSID=1467_21107_29522_29521_29237_28519_29098_28838_29220; BD_UPN=12314753
```

响应body

```
{"errno":0,"request_id":"3376203309","timestamp":1562943376,"data":[]}
```



分类	分类描述
1xx	信息，服务器收到请求，需要请求者继续执行操作
2xx	成功，操作被成功接收并处理
3xx	重定向，需要进一步的操作以完成请求
4xx	客户端错误，请求包含语法错误或无法完成请求
5xx	服务器错误，服务器在处理请求的过程中发生了错误



HTTP异常状态码解决方案

状态码	排查思路
400 (Bad Request/错误请求)	检查脚本中的请求参数语法格式是否正确
401 (Unauthorized/未授权)	检查脚本中的请求是否缺少必要的header信息
403 (Forbidden/禁止)	检查脚本中的用户信息，是否拥有权限去操作业务
404 (Not Found/未找到)	1、检查请求url地址，或者url中拼接的参数是否正确 2、检查web服务器是否成功启动，如查看tomcat日志
405 (Method Not Allowed/方法未允许)	1、检查http请求方式是否正确，比如GET请求写成POST方式 2、检查请求参数是否正确
500 (Internal Server Error/内部服务器错误)	服务端程序内部报错，首先登录服务器查看运行日志，根据报错原因排查问题，可能有两方面原因 1、代码逻辑错误，造成报错 2、请求参数异常，造成服务端无法处理
502/503/504 Bad Gateway/错误的网关 Service Unavailable/服务无法获得 Gateway Timeout/网关超时	1、如果单次调用接口就报该错误，说明是后端服务器配置有问题，或者服务不可用 2、如果并发压测时出现此错误，说明是后端压力太大，出现异常，此问题一般是后端出现了响应时间过长或者无响应造成的



浏览器

Chrome/Firefox: F12

APP

Android/iOS: Fiddler/Charles

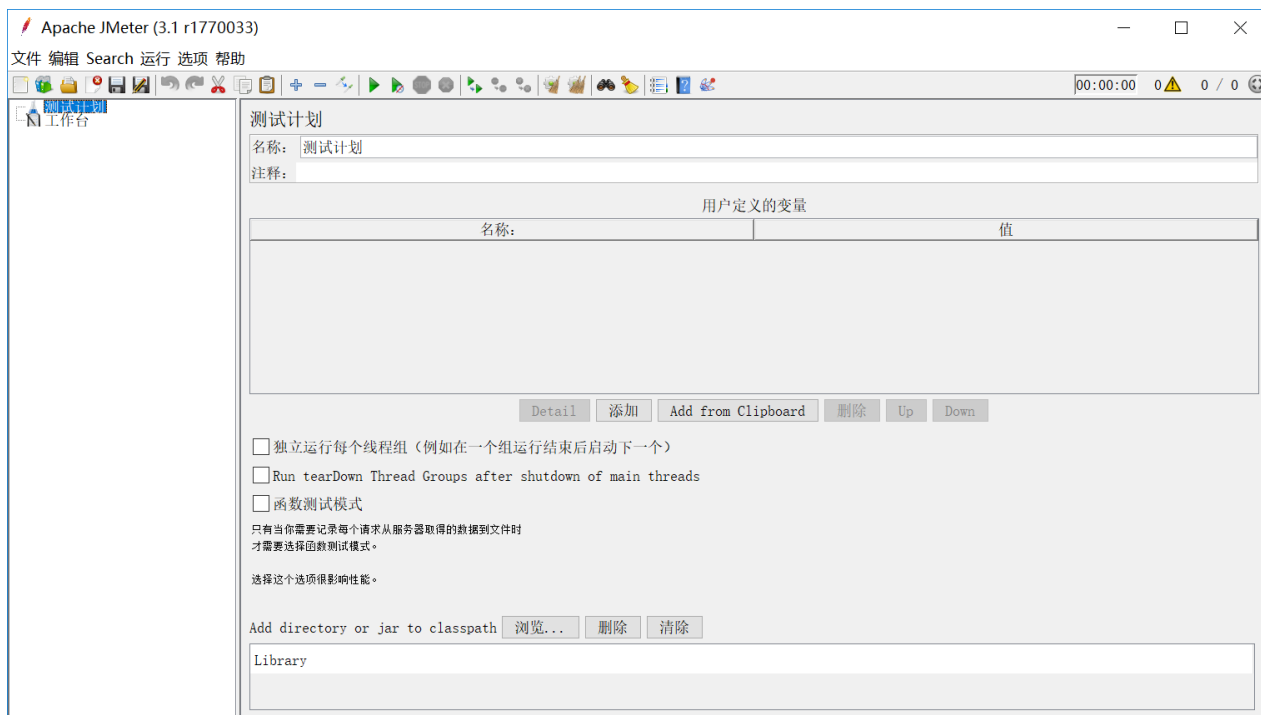


JMeter, 一个100%的纯Java桌面应用, 由Apache组织的开放源代码项目, 它是功能和性能测试的工具。具有高可扩展性、支持Web(HTTP/HTTPS)、SOAP、FTP、JAVA等多种协议的特点。

官方网站: <https://jmeter.apache.org/>



- 1、本机配置好Java环境变量
- 2、官网下载压缩包，在任意目录下解压
- 3、修改配置文件，打开Jmeter的bin目录下jmeter.properties，修改
`language=zh_CN`
`sampleresult.default.encoding=utf-8`
- 4、进入到Jmeter的bin目录下，双击jmeter.bat启动

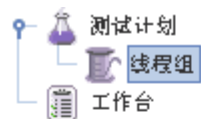




- 测试计划
- 线程组
- 采样器 (sampler)
- 断言
- 监听器



添加线程组并设置参数大小：测试计划→添加→Threads →线程组



线程组 给线程组起名字

名称：线程组

注释：

在取样器错误后要执行的动作

☒ 继续 ☐ Start Next Thread Loop ☐

线程属性 设置10个并发

线程数：10

Ramp-Up Period (in seconds): 0 所有线程在多少时间内启动，如果设置5，那么每秒启动2个线程

循环次数 ☒ 永远 请求的重复次数，如果勾选“永远”，将一直发送请求

☐ Delay Thread creation until needed

☒ 调度器 设置启动时间和结束时间，优先级分别低于启动延迟和持续时间，会被启动延迟和持续时间覆盖

调度器配置

启动时间 2014/06/06 14:47:59

结束时间 2014/06/06 14:47:59 设置场景运行的时间：10分钟

持续时间（秒） 600

启动延迟（秒） 设置场景延迟启动的时间



- Get接口
- POST接口 (key=value)
- POST接口2 (json)
 - 需要增加header (Content-type:application/json)
- POST接口3 (key=json)

注：各接口url见pinter项目接口文档



HTTP请求

名称: HTTP请求

注释:

基本 高级

客户端实现

实现: ▼

超时 (毫秒)

连接: 5000 响应: 5000

从HTML文件嵌入资源

☐ 从HTML文件获取所有内含的资源 ☐ 并行下载. 数量: 6 网址必须匹配:

源地址

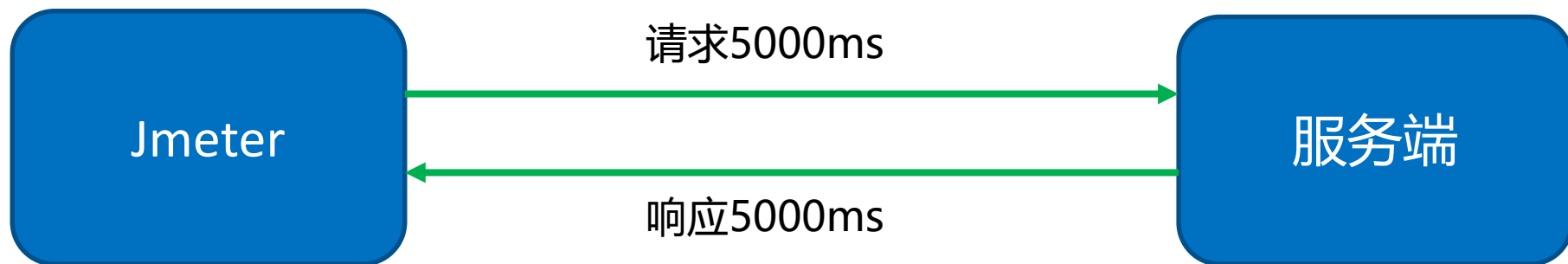
IP/主机名 ▼

代理服务器

Scheme: 服务器名称或IP: 端口号: 用户名: 密码:

其他任务

☐ 保存响应为MD5哈希





添加断言-JSON断言

JSON断言

名称: JSON断言

注释:

Assert JSON Path exists: \$.code

☒ Additionally assert value

☒ Match as regular expression

Expected Value:

q|

☐ Expect null

☐ Invert assertion (will fail if above conditions met)

json路径表达式

\$.code 代表json中的code字段值

详细用法参见: <https://github.com/json-path/JsonPath>



添加查看结果树：监听器-查看结果树

察看结果树

名称: 察看结果树

注释:

所有数据写入一个文件

文件名

显示日志内容: ☐ 仅错误日志 ☐ 仅成功日志

查找: ☐ 区分大小写 ☐ 正则表达式

Text

✓ HTTP请求

取样器结果 请求 响应数据

Response Body Response headers

Find ☐ 区分大小写 ☐ 正则表达式

```
{"code": "0", "message": "success", "data": {"skuId": 1, "skuName": "ptest-1", "price": "28", "stock": 876, "brand": "testfan"}}
```



CSV文件读取: `${_CSVRead(D:\data.txt,0,)}`

随机数: `${_Random(1,100,)}`

随机字符串: `${_RandomString(8,abcdefghijklmnopqrstuvwxyz0123456789,)}`

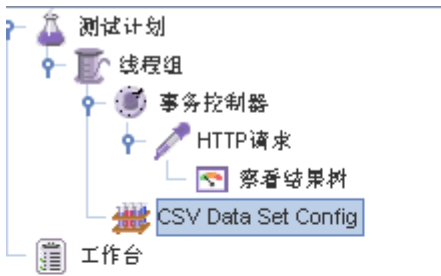
时间戳: `${_time(,)}`

生成唯一UUID: `${_UUID}`

注: 上述函数都可以将结果保存到一个变量里, 函数的最后一个参数为变量名称



添加CSV Data Set Config：线程组→添加→配置元件 → CSV Data Set Config



CSV Data Set Config

名称： CSV Data Set Config

注释：

Configure the CSV Data Source

Filename:	C:\canshuhua1.txt	参数化要引用的文件名和路径
File encoding:		文件编码，可以不填
Variable Names (comma-delimited):	id	自定义变量名(多个变量时用逗号分隔)
Delimiter (use "t" for tab):	,	参数文件中多列变量值的分隔符，默认是逗号
Allow quoted data?:	False	选true时读取中文参数值出现乱码
Recycle on EOF ?:	True	是否循环读入
Stop thread on EOF ?:	False	是否停止线程
Sharing mode:	All threads	共享模式，分三种不同取值方式

Recycle on EOF	Stop thread on EOF	参数效果
True	False	重复
False	True	唯一



Throughput

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

对于接口测试来说, Jmeter里的吞吐量=TPS

聚合报告

名称: 聚合报告

注释:

所有数据写入一个文件

文件名: 浏览... Log/Display Only: ☐ 仅日志错误 ☐ Successes

Label	# Samples	Average	Median	90% Line	Min	Max	Error %	Throughput	KB/sec
线程组:事...	9871	2	3	3	2	6	0.00%	329.1/sec	4674.7
总体	9871	2	3	3	2	6	0.00%	329.1/sec	4674.7

#Samples: 表示一共发出了多少个请求

Average: 平均响应时间

Median: 中位数, 类似于50% Line

90% Line: 假设该值为20, 那么90%用户的响应时间小于等于20ms

Min: 最小响应时间

Max: 最大响应时间

Error%: 错误率, 出现错误的请求数量/请求的总数

Throughput: 吞吐量

KB/Sec: 每秒从服务器端接收到的数据量

☒ Include group name in label? ☒ Save Table Header

保存结果



使用Jmeter插件可以扩展Jmeter的功能

插件官网: <http://jmeter-plugins.org/downloads/all>

使用Jmeter插件管理器, 可以自动下载并安装插件, 更加方便简单

几个好用的插件:

- 1> 3 Basic Graph: windows下可用的实时tps和响应时间的插件
- 2> Custom JMeter Functions 扩展函数
- 3> Random CSV Data Set Config 随机csv文件参数化
- 4> PerfMon 服务端性能实时监控插件



常用的逻辑控制器

- 1、循环控制器：可以设置该控制器内的sampler执行的次数，循环次数与线程的循环次数各自独立
- 2、if控制器：根据判断条件决定是否执行该控制器内的请求，如果是字符串比较条件，参数和字符串都需要加引号

条件格式：

`${_jexl3(条件表达式)}`

如：`${_jexl3(${num} > 10)}`、`${_jexl3("${num}" == "10")}`

- 3、仅一次控制器：该控制器内的请求只执行一次，无论线程循环多少次
- 4、foreach控制器，可以遍历某个参数数组，循环获取数组中的参数



常用的后置处理器

JSON提取器

json路径表达式: <https://github.com/json-path/JsonPath>

正则表达式提取器

三步走

- 1、拷贝目标数据和左右边界
- 2、把目标数据用括号括起来
- 3、把目标数据用.+?代替



调试取样器可以打印出来Jmeter运行过程中保存下来的参数，需要配合查看结果树使用

调试取样器

名称:

调试取样器

注释:

JMeter 属性:

False

▼

JMeter 变量:

True

▼

系统属性:

False

▼



固定定时器

设置一个固定的sleep时间

同步定时器

Jmeter里的集合点，并发会在此停留，等待指定的线程数达到时，再进行下一步操作

常量定时器

控制请求的TPS，按照分钟控制，比如设置为6000，就是控制最高TPS为 $6000/60=100$



上传:

- POST请求, 勾选 use ...for post
- 同请求一起发送文件里, 填写文件名称, 参数名称
- MIME类型: application/octet-stream

下载:

普通GET请求

练习案例:

参见pinter项目首页链接



http cookie管理器

可以在浏览器中抓取到cookie信息，然后通过http cookie管理器为http请求添加cookie信息

存储在Cookie管理器中的Cookie				
名称:	值	域	路径:	安全
JSESSIONID	9FFC7DDB4E6EEE33D56C64...			<input type="checkbox"/>

Jmeter的cookie处理机制

和Loadrunner一样，Jmeter可以自动处理cookie，但是需要在线程组内添加一个空的cookie管理器

练习案例：见pinter项目首页链接



http信息头管理器

在http信息头管理器中添加一个或多个http请求header中的名称和值，可以修改请求头的值

HTTP信息头管理器

名称: HTTP信息头管理器

注释:

信息头存储在信息头管理器中

名称:	值
Cache-Control	max-age=1

练习案例

参加pinter项目首页链接



关键点:

- 1、添加http post请求
- 2、添加header: Content-type:text/xml

参考接口: <http://ws.webxml.com.cn/WebServices/MobileCodeWS.asmx>



很多HTTP接口在传参时，需要先对接口的参数进行数据签名加密

如pinter项目中的签名接口

<http://localhost:8080/pinter/com/userInfo>

参数为：

```
{"phoneNum":"123434","optCode":"testfan","timestamp":"1211212","sign":"fdsfdsaafsasfas"}
```

其中，sign字段是按照特定算法进行加密后的数据

本接口的签名算法为

$\text{sign} = \text{Md5}(\text{phoneNum} + \text{optCode} + \text{timestamp})$



添加数据库jar包

拷贝mysql驱动包到jmeter/lib目录下

或者在测试计划处添加驱动jar包



配置数据库链接信息

添加JDBC Connection Configuration

需要配置项

Variable Name Bound to Pool	
Variable Name for created pool:	my_db_config

Database Connection Configuration	
Database URL:	jdbc:mysql://localhost:3306/oa?useUnicode=true&characterEncoding=utf8
JDBC Driver class:	com.mysql.jdbc.Driver
Username:	root
Password:	●●●●●●

URL: jdbc:mysql://{ip}:{port}/{dbname}?useUnicode=true&characterEncoding=utf8



创建JDBC sampler

SQL语句类型

select

update

执行方式分类

普通SQL

预编译SQL: sql语句中的数值用? 代替

Parameter values: 代替sql语句中的问号的数据

Parameter types: 数据库字段数据类型, 如VARCHAR,INTEGER

Variable Names: 变量名称, 跟SQL语句中查询字段数量保持一致



- BeanShell是一种完全符合Java语法规范的脚本语言,并且又拥有自己的一些语法和方法;
- BeanShell是一种松散类型的脚本语言(这点和JS类似);
- BeanShell是用Java写成的,一个小型的、免费的、可以下载的、嵌入式的Java源代码解释器,具有对象脚本语言特性,非常精简的解释器jar文件大小为175k。
- BeanShell执行标准Java语句和表达式,另外包括一些脚本命令和语法。



- 前置处理器: BeanShell PreProcessor
- 采样器: BeanShell Sampler
- 后置处理器: BeanShell PostProcessor
- 断言: BeanShell断言



内置变量

beanshell脚本中不用定义，可以直接使用的变量，常用的内置变量和方法如下

log: 写日志到控制台和jmeter.log，如log.info("xxxxx");

vars: 操作jmeter变量

vars.get("skuld");从jmeter中获取\${skuld}变量的值

vars.put("name" ," test");将" test" 保存到\${name}变量中

prev: 获取前面sampler返回的信息

getResponseDataAsString(): 获取响应信息

getResponseCode(): 获取响应code

更多内置变量参考: <https://jmeter.apache.org>



BeanShell面板上写脚本

需求：

- 1、调用接口获取sku信息
- 2、判断库存，如果库存大于500，调用buy接口购买10个商品，否则购买5个商品

// 获取接口返回的库存值

```
String myStock = vars.get("p_stock");
```

// 转换为整数

```
int iStock = Integer.parseInt(myStock);
```

// 判断库存

```
if (iStock > 500){
```

 // 重新保存参数

```
    vars.put("buyNum","10");
```

```
}else{
```

```
    vars.put("buyNum","5");
```

```
}
```



引用外部java源文件

// 引入源代码

```
source("D:/Md5Util.java");
```

// 生成随机手机号

```
String phone = "135${__Random(10000000,99999999,myPhone)}";
```

```
String code = "testfan";
```

// 生成时间戳

```
String time = "${__time(,myTime)}";
```

// 调用外部函数进行加密

```
String md5 = Md5Util.getMd5Hex(phone+code+time);
```

// 将数据另存为新的变量

```
vars.put("phone",phone);
```

```
vars.put("md5",md5);
```



调用jar包

- 1、测试计划，Add directory or jar to classpath
- 2、import 所需要的类名

```
import com.lee.util.Md5Util
```

```
// 生成随机手机号
```

```
String phone = "135${__Random(10000000,99999999,myPhone)}";
```

```
String code = "testfan";
```

```
// 生成时间戳
```

```
String time = "${__time(,myTime)}";
```

```
// 调用外部函数进行加密
```

```
String md5 = Md5Util.getMd5Hex(phone+code+time);
```

```
// 将数据另存为新的变量
```

```
vars.put("phone",phone);
```

```
vars.put("md5",md5);
```



内置变量

Failure: 是否失败, **boolean**类型

FailureMessage: 失败日志, 在断言失败时显示

```
int iStock = Integer.parseInt(vars.get("p_stock"));
if (iStock > 1500){
    Failure = true;
    FailureMessage = "库存数量超过了1500";
    // ResponseData是服务器返回的byte[]类型的数据
    // 如果想打印, 必须转换为String类型的, 用new String(ResponseData)
    log.info(new String(ResponseData));
    //打印当前请求的url, SamplerData是String类型的数据
    log.info(SamplerData);
}
```




需求

- 1、调用登录接口，获取token值
- 2、将token值保存到一个文件里

```
String line = vars.get("userName")+","+vars.get("token");
try {
    BufferedWriter writer = new BufferedWriter(new FileWriter("D:\\output.txt",true));
    writer.write(line);
    writer.newLine();
    writer.close();
} catch (IOException e) {
    e.printStackTrace();
}
```



准备工作:

- 1, 压力机安装并配置好JDK
- 2、配置jmeter的环境变量
- 3, 调试好jmeter脚本

单机器测试步骤:

执行 `jmeter -n -t test.jmx -l result.jtl`

-n: 命令行模式, no-gui

-t: jmx脚本路径;

-l: jtl结果文件存放路径



概要日志数据解释

```
summary + 444851 in 6.4s = 69248.3/s Avg: 0 Min: 0 Max: 10 Err: 0 (0.00%) Active: 64 Started: 64 Finished: 0
summary + 210979 in 30s = 70267.0/s Avg: 0 Min: 0 Max: 7 Err: 0 (0.00%) Active: 64 Started: 64 Finished: 0
summary = 2552930 in 36.4s = 70089.2/s Avg: 0 Min: 0 Max: 10 Err: 0 (0.00%) Active: 64 Started: 64 Finished: 0
summary + 2136879 in 30s = 71226.9/s Avg: 0 Min: 0 Max: 9 Err: 0 (0.00%) Active: 64 Started: 64 Finished: 0
summary = 4689809 in 66.4s = 70604.1/s Avg: 0 Min: 0 Max: 10 Err: 0 (0.00%) Active: 64 Started: 64 Finished: 0
summary + 2217377 in 30s = 73910.1/s Avg: 0 Min: 0 Max: 9 Err: 0 (0.00%) Active: 64 Started: 64 Finished: 0
summary = 6907186 in 96.4s = 71633.5/s Avg: 0 Min: 0 Max: 10 Err: 0 (0.00%) Active: 64 Started: 64 Finished: 0
summary + 2212605 in 30s = 73751.0/s Avg: 0 Min: 0 Max: 10 Err: 0 (0.00%) Active: 64 Started: 64 Finished: 0
summary = 9119791 in 126s = 72385.6/s Avg: 0 Min: 0 Max: 10 Err: 0 (0.00%) Active: 64 Started: 64 Finished: 0
summary + 2201514 in 30s = 73381.4/s Avg: 0 Min: 0 Max: 11 Err: 0 (0.00%) Active: 64 Started: 64 Finished: 0
summary = 11321305 in 156s = 72375.8/s Avg: 0 Min: 0 Max: 11 Err: 0 (0.00%) Active: 64 Started: 64 Finished: 0
```

最近30s TPS →

平均TPS →

最大响应时间

失败交易和失败率

并发数



三种方式来获取Jmeter的结果报表

一、在GUI模式下跑Jmeter的脚本，用tps插件实时展示图表

二、在命令行模式下跑Jmeter的脚本，生成的jtl文件，在GUI界面的聚合报告里打开，可以展示tps和 响应时间等数据

三、在命令行模式下跑Jmeter的脚本，生成的jtl文件，通过Jmeter自带命令，生成html报表

注意：

在实际工作中，不推荐第一种执行方式，会影响Jmeter的性能。



Html报表生成步骤:

- 1, 进入jmeter的bin目录下, 修改reportgenerator.properties
- 2, 修改jmeter.reportgenerator.overall_granularity=1000 (报表中数据展示间隔1秒)
- 3, 创建一个存放数据报表的文件夹
- 4, 执行命令: `jmeter -g result.jtl -o ./output`

其中:

- g 指定jtl文件的路径
- o 指定html报表生成到哪个文件夹下

注意: 只有Jmeter3.0版本以上支持此功能



Jmeter两种运行模式

- 按照运行次数运行：线程组设置循环x次
- 按照运行时间运行
 - 1> 线程组设置循环永远
 - 2> 勾选调度器，设置持续时间，单位秒



关键逻辑

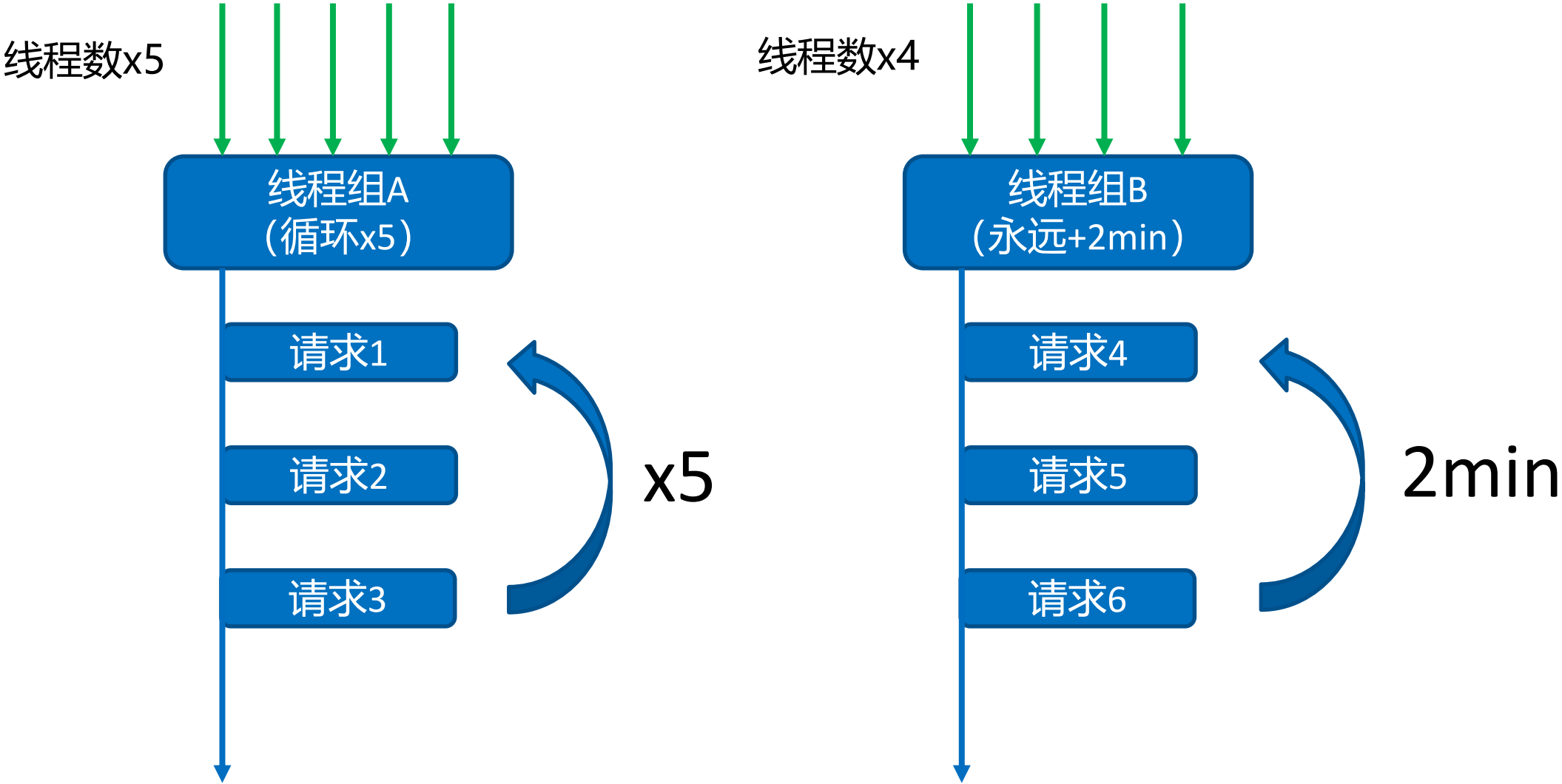
- 一个线程组内的多个请求是顺序执行的
- 不同线程组内的请求是并行执行的

实际工作中如何设置线程组和请求的关系？

- 如果多个接口之间没有强依赖关系，每个接口单独设置一个线程组
- 如果多个接口之间有依赖关系（数据关联），把有关联关系的接口按照顺序放在同一个线程组内



线程组和请求的运行逻辑





setUp线程组

执行全局初始化操作的线程组，类似LR脚本中的init函数

tearDown线程组

执行全局结束操作的线程组，类似LR脚本中的end函数



需求

对pinter项目案例三接口进行单接口压测



需求

对pinter项目案例四接口进行单接口压测



需求

对pinter项目案例三和案例四接口进行混合压测，比例1:2

Thanks

