# 性能测试第二天课堂笔记

# 昨日回顾:

(1) 性能测试的概述

性能测试的关注点: 时间和资源

性能测试的概念:使用自动化的工具,模拟用户真实的使用场景,对软件各项性能指标进行测试和评估

的活动

性能测试的目的:评估当前的系统能力; 出现性能bug后,优化性能;预测未来的性能需求是否满足

性能和功能的对比:

• 焦点:功能(正向、逆向),性能(时间、资源)

• 关联: 先功能测试, 再性能测试

(2) 性能测试的策略

#### 基准测试:

• 狭义上: 单用户测试, 记录性能指标

• 广义上:设定基准线,后续软硬件条件发生变化时,再进行同样基准测试对比观察性能的变化。

**负载测试:逐步增加**系统负载,找到满足系统需求情况下的最大负载量的测试。

稳定性测试:在用户正常的业务负载下,长时间(1天-7天)测试,观察系统是否能稳定运行

并发测试: 极短时间内, 发送大量请求, 观察系统是否能并发执行

压力测试:在高负载的情况下,观察系统是否有好的容错能力和可恢复能力。包括:高负载下的长时间

稳定性压力测试 和 极限负载下的破坏力压力测试。

(3) 性能测试的指标

响应时间:客户端发起请求开始,到收到响应的总时间。包含:服务器处理时间+网络传输时间

并发(用户)数:同一段时间往服务器发送请求的用户数

吞吐量:单位时间内,服务器处理的客户端请求的数量。QPS和TPS

点击数:访问页面时,加载页面的各种元素(文本、图片、css、is等)的请求数量。

错误率:在用户负载情况下,失败业务的概率。注意:与功能的随机bug区分

资源使用率:使用系统时,资源占用的比例。常见:CPU、内存、磁盘、网络

(4) 性能测试的流程

性能需求分析

性能测试计划和方案

性能测试用例

性能测试执行: 搭建性能环境、准备测试脚本、配置监控指标、执行测试脚本

性能结果分析和调优

性能测试的报告

# 当日学习目标

- 掌握Jmeter环境搭建
- 理解Jmeter的主要元件及相关的作用域
- 理解线程组、HTTP请求的配置使用
- 掌握Jmeter参数化的应用

# 性能测试工具

## JMeter和Loadrunner工具对比:

- (1) 主流性能测试工具Loadrunner和Jmeter对比 相同点
- 都能模拟大量用户
- 都能支持多协议 (常见的协议都支持,如: HTTP)
- 都有监控及分析报表功能
  - (2) 主流性能测试工具Loadrunner和Jmeter对比 不同点

工具	用户量	分析报表	IP欺骗功能	费用	体积	扩展性
Loadrunner	多 (万)	精确 (秒)	支持	收费	大 (单位GB)	不能扩展
Jmeter	少	较差 (分钟)	不支持	免费	小 (50MB)	有扩展组件

结论:项目日常性能测试Jmeter足够用,出商业报告优先Loadrunner

## jmeter环境搭建(重点):

#### 安装JDK:

•下载JDK - 安装JDK - 配置环境变量 - 验证

#### 安装|meter:

•下载|meter - 安装|meter - 配置环境变量 - 启动验证

#### 注意点:

- •下载JDK时,注意电脑操作系统是32位/64位
- •下载Jmeter时,注意与本机安装的JDK版本匹配
- ·安装Jmeter时,安装路径中不能有中文/空格

# jmeter的功能概要:

## jmeter文件目录结构:

• 修改JMeter配置文件 — Bin目录

• 下载第三方插件 (jar包) 并使用 — lib/ext目录

• 查找用户帮助手册 — printable docs目录

• 启动Jmeter程序 — Bin目录

## 基本配置

## Jmeter界面汉化:

• 永久性: 修改配置文件, language=zh CN, 重启Jmeter

• 临时性: 菜单 'Options'->Choose Language->Chinese (Simplified)

## Jmeter主题修改:

• 菜单、选项′-> 外观-> Windows (选择自己喜欢的主题即可)

# jmeter元件及基本作用域(理解):

### 基本元件:



#### (1) 如下接口自动化脚本的实现过程对应着Jmeter哪个元件?

1. 初始化测试数据 — 配置元件

2. 对请求参数化进行赋值 — 前置处理器

3. 调用GET/POST方法发送请求 — 取样器

4. 提取响应中特定字段的值 — — 后置处理程器

5. 对提取出来的值与预期结果进行对比 — 断言

#### (2) 元件与组件有什么关系?

元件:多个类似功能组件的<mark>容器</mark>(类似于类)

组件:容器中实现独立的某个功能(类似于方法)

## 作用域的原则:

Ø取样器:核心,没有作用域

Ø逻辑控制器:只对其子节点中的取样器和逻辑控制器起作用

Ø其他元件:

•如果是某个取样器的子节点,则该元件只对其父节点起作用

•如果其父节点不是取样器,则其作用域是该元件父节点下的其他所有后代节点(包括子节点,子节点的子节点等)

## 元件执行顺序:

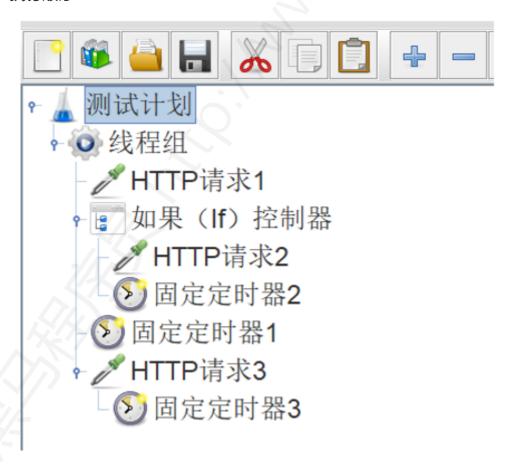
### 在同一个作用域(目录/级别/缩进)的不同元件的执行顺序:

配置元件 - 前置处理程序 - 定时器 - 取样器 - 后置处理程序 - 断言 - 监听器

#### 在同一个作用域(目录/级别/缩进)的相同元件的执行顺序:

从上到下的顺序依次执行

### 案例: 执行顺序:



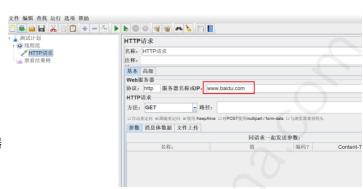
正确: 定时器1 - 请求1 - 定时器1 - 定时器2 - 请求2 - 定时器1 - 定时器3 - 请求3

# Jmeter第一个案例:

需求:使用JMeter访问百度首页接口,并查看请求和响应信息

#### 步骤:

- 1. 启动JMeter
- 2. 在`测试计划'下添加`线程组'
- 3. 在`线程组'下添加`HTTP请求'取样器
- 4. 填写 \HTTP请求'的相关请求数据
- 5. 在 '线程组'下添加 '察看结果树' 监听器
- 6. 点击 '启动' 按钮运行, 并查看结果



# Jmeter三个重要组件(重点):

### 线程组的介绍:

### 特点:

•控制」meter用于执行测试的一组用户

## 线程组的分类:

·Setup线程组:预测试操作,所有脚本之前执行

•普通线程组: 执行测试用例,可以有1个或者多个 (并行/串行)

•Teardown线程组:测试后操作,所有脚本之后执行

## 线程组的属性:

参数介绍:

	<b>?</b>	00
线程组		
名称: 线程组		
注释:		
在取样器错误后要执行的动	乍————	
	◉继续 ○启动下一进程循环 ○	停止线程 ○停止测试 ○立即停止测试
线程属性		
线程数: 1	虚拟用户数	100
Ramp-Up时间(秒): 1	全部虚拟用户数启动需要的时	间 5
循环次数□永远 1	指定运行次数,选中"永远"后	一直运行。
□延迟创趸线程直到需要	默认启动即创建所有线程; 勾	〕选则运行对应线程时才创建。 (对外无体现)
□调度器配合使用	勾选后才能配置"持续时间"和	口"启动延迟"。循环次数设置成"永远"时调度器才能生效
调度器配置		
持续时间(秒)	设置脚本持续运行时间	
启动延迟(秒)	脚本启动延迟等待的时间	

## (3) 如下场景如何设置线程组?

• 模拟10个用户并行执行: —— <del>线程数</del>

• 模拟10个用户5s内启动完成: —— 线程数10, ramp-up时间: 5s

• 模拟2个用户各循环3次: —— <del>线程数: 2, 循环次数: 3</del>

• 模拟2个用户运行30s: —— 线程数: 2, 循环: 永远, 持续时间: 30s

• 模拟2个用户等待10s后开始执行: ——在上一个的基础上,增加延迟启动时间: 10s

#### 案例分析:

- (5) 使用1个线程组,添加HTTP请求 (百度)
- 配置线程数为2,循环次数为3时,运行观察结果
- 配置线程数为3,循环次数为2时,运行观察结果,对比不同

#### 分析:

- 线程数代表虚拟用户数,用户数越多,负载越大
- 循环次数代表运行时间,次数越多,运行时间越长

### HTTP请求:

### 参数介绍:

作用: 向服务器发送http及https请求

位置: 选中线程组->右键->添加->取样器->HTTP请求

参数:

HTTP请求				
名称: HTTP请求				
主譯:				
AV Ab the ter				
基本。高级				请求行
Web服务器				
<b>HTTP (默认) /HTTPS</b> 服务器名称或P:	www.baidu.com 服务器域名或者服务器	IP地址	<sup>                                    </sup>	10号
HTTP请求				
方法: GET /post/ v 篇位: path,	即URL端口后面的部分			内容编码: utf-8
□ 目动重定问 ☑ 跟随重定问 ☑ 使用 KeepAlive	□ 対POST使用multipart / form-data	□ 与浏览器兼容的头		
参数   消息体数据   文件上传				
	阿请求一起发送	<b>参数:</b>		
名称:	值	编码?	Content-Type	包含等于?
get (URL路径中传参数)	: ?犍名=犍值&犍名=键值	NOST (请求体中		
1、在path路径后加参数。 2、在参数列表中添加参数	?键名=键值&键名=键值		数据中添加请求体(任意	
、 2、任参数列表中添加参数		2、如果消水仲中	为form表单格式,可以再	多数列表中添加
			- Value de la constant de la constan	AL CHEALS WE
			请求	体 (请求参数)
	详细 添加 从剪贴板添加	1 删除 向上 向	下	

#### 案例一(使用HTTP请求路径来传递get请求参数):

- (1) 使用1个线程组,添加HTTP请求(百度),路径为:/S
- 使用HTTP协议, GET请求方法, 路径中添加参数为: wd = test, 运行观察结果

#### 案例二 (使用参数列表来传递get请求的参数):

• 使用HTTPS协议,GET请求方法,路径中添加参数为: wd = test,运行观察结果

#### 案例三(使用参数列表来传递POST请求的form格式参数):

● 使用HTTPS协议,POST请求方法,消息体数据中添加参数为:wd=test,运行观察结果

#### 案例四:

• 使用HTTP协议, POST请求方法, 消息体数据中添加参数为: wd = test, 运行观察结果

/ 05 HTTP参数介绍.jmx (D:\传智播客\02 授课\上海23期-性能测试\day2\file\05 HTTP参数介绍.jmx) - Apache JMeter (5.0 r1840935)

文件 编辑 查找 运行 选项 帮助 🕶 🗼 测试计划 HTTP请求 💉 察看结果树 名称: HTTP请求1 🕶 线程组 - 百度1 \_ get路径传递参数 MTTP请求1 注释: ◎ 线程组 \_ 百度2\_ get 参数列表传递参数 基本 高级 // HTTP请求2 Web服务器 ◎ 线程组 \_ 百度3\_ post参数列表来传递 协议: http 服务器名称或IP: www.baidu.com // HTTP请求3 ◉ 线程组 \_ 百度4 \_ post消息体数据传递参 HTTP请求 》HTTP请求4 方法: GET ▼ 路径: /S?wd=test □自动重定向 ☑跟随重定向 ☑使用 KeepAlive □对POST使用multipart / form-data □ □ 参数「消息体数据「文件上传 同请求一起发送参 名称:

## 查看结果树:

各部分的含义:

# (1) 如何查看HTTP消息请求和响应内容?

# 如何查看请求参数:

• 请求 - Request Body (请求行 + 请求体)

# 如何查看响应结果:

- 响应数据 Response Body (响应体)
- (2) 如何解决Jmeter查看结果树的响应中的中文乱码?

# 修改配置文件

- 1. 找到jMeter安装目录下的bin目录
- 2. 打开jmeter.properties文件, 修改配置 sampleresult.default.encoding=UTF-8
- 3. 重启JMeter即可

# Jmeter参数化 (重点)

定义: 使用不同的测试数据, 调用相同的测试方法进行测试

本质: 实现测试数据与测试方法的分离。

#### 实现方式:

- 用户定义的变量 —— 全局变量
- 用户参数 —— 为每个用户分配不同的参数值
- CSV数据文件设置 —— 文件方式参数化
- 函数 —— 随机数据
- 数据库

## 用户定义的变量:

# (1) 什么时候使用用户定义的变量?

定义全局变量

# (2) 使用"用户定义的变量"进行参数化的步骤?

- 1.添加线程组
- 2.添加用户定义的变量。格式:变量名 变量值
- 3. 添加HTTP请求, **引用定义的变量名**。格式: \${**变量名**}
- 4. 添加查看结果树

## 用户参数:

## (1) 什么时候使用用户参数?

• 针对同一组参数, 当不同的用户来访问时, 可以获取到不同的值

## (2) 使用"用户参数"进行参数化的步骤?

1.添加线程组,设置线程数为n (表示模拟的田户数)

- 2.添加用户参数
  - 第一列添加多个变量名
  - 后续每一列为一组用户的数据
- 3.添加HTTP请求,引用定义的变量名。格式: \${变量名}
- 4.添加查看结果树

## CSV数据文件设置:

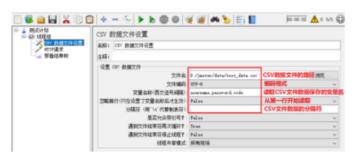
# (1) 什么时候使用CSV数据文件设置?

• 当不同的用户,或者同一个用户多次循环时,都可以获取到不同的值

# (2) 使用"CSV数据文件设置"讲行参数化的步骤?

- 1. 定义CSV数据文件
- 2. 添加线程组
- test\_data.csv

  1 user01,123456,0000
  2 user02,123456,1111
  3 user03,123456,2222
- 3.添加CSV数据文件设置



- 4. 添加HTTP请求, **引用定义的变量名**。格式: \${**变量名**}
- 5. 添加查看结果树

### 函数:

# (1) **什么时候使用\_\_counter函数?**

• 自动生成不重复的数据,让每个用户每次循环都能取到不同的数据,且不需要提前定义

## (2) 使用"counter函数"进行参数化的步骤?

- 1. 添加线程组,设置虚拟用户数和循环次数
- 2. 生成 counter函数



- 3. 添加HTTP请求,使用\_\_counter函数。格式: \${\_\_counter(FALSE,)}
- 4. 添加查看结果树

#### 四种参数化方式的对比:

## (1) 什么是参数化?

把测试数据组织起来,用**不同的测试数据**调用相同的测试方法。

## (2) 4种参数化方式有何不同? 如何选择适当的方式?

### 用户定义的变量:

- 作用: 定义全局变量

- 局限性: 每次取值 (无论是否相同的用户) 都是固定值

### 用户参数:

- 作用: 保证不同的用户针对同一组参数, 可以取到不同的值

- 局限性: 同一个用户在多次循环时, 取到相同的值

### CSV数据文件设置:

- 作用: 保证不同的用户及同一用户多次循环时, 都可以取到不同的值

- 局限性: 需要手动进行测试数据的设置

### 函数:

- 作用: 保证不同的用户及多次循环时,都可以取到不同的值,不需要提前设置

- 局限性: 输入数据有特定的业务要求时无法使用(如: 登录时的用户名密码)