

性能测试

--性能测试基础知识2

知识点回顾

- 什么是性能测试
- 为什么做性能测试
- 性能测试怎样做
 - 性能测试流程
 - 性能测试主要术语

HTTP协议的主要特点

1 支持客户/服务器模式

2 简单快捷

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、HEAD、POST等。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

HTTP协议主要特点

- **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
- **无连接：**
 - 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他的应答就较快

HTTP协议特点

■ Keep Alive

- 长连接
- 短连接

HTTP(Hyper Text Transfer Protocol)

■ 什么是HTTP?

- 是一种按照URL指示，将超文本文档从一台主机(Web服务器)传输到另一台主机(浏览器)的应用层协议，以实现超链接的功能

HTTP协议详解

■ HTTP请求

— 请求行

- 请求方法、URL、协议版本等请求头（消息报头）

— 请求头

- 由一个头域名、冒号和值域组成

— 请求正文

- 也叫请求数据，在使用POST请求提交表单数据的时候，这些表单数据就会被放在HTTP请求的请求正文中，以加密的形式向服务器传输

HTTP协议详解

▼ Request Headers

view parsed

GET / HTTP/1.1

请求行

Host: www.itxdl.cn

Connection: keep-alive

请求头

Cache-Control: max-age=0

Upgrade-Insecure-Requests: 1

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.86 Safari/537.36

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8

Accept-Encoding: gzip, deflate

Accept-Language: zh-CN,zh;q=0.8,en;q=0.6

Cookie: UM_distinctid=15cae6aef3f2a-06f8414ac81c3E

HTTP请求方法

■ 主要请求方法GET 和 POST

- GET方式：用于获取请求页面指定信息，主要用于获取网络资源
- POST方式：用于向服务器发送大量数据，主要用于表单提交

HTTP请求协议

- **HEAD** 请求获取由Request-URI所标识的资源的响应消息报头
- **PUT** 请求服务器存储一个资源，并用Request-URI作为其标识
- **DELETE** 请求服务器删除Request-URI所标识的资源
- **TRACE** 请求服务器回送收到的请求信息，主要用于测试或诊断
- **CONNECT** 保留将来使用
- **OPTIONS** 请求查询服务器的性能，或者查询与资源相关的选项和需求

HTTP响应

■ 响应分为三部分

— 响应行

- 协议和状态码

— 状态码

1. **1xx**: 信息提示, 表示请求已被成功接收, 继续处理。范围: 100~101
2. **2xx**: 服务器成功处理了请求。范围: 200~206
3. **3xx**: 重定向。访问资源被移动, 告诉客户端新资源地址, 浏览器重新对信资源发起请求。范围: 300~305
4. **4xx**: 客户端错误。如: 格式错误, URL不存在等。范围: 400~415
5. **5xx**: 服务器错误。服务器内部错误。范围: 500~505

HTTP响应

- 响应头
- 响应正文
 - 服务器向客户端发送的HTML数据

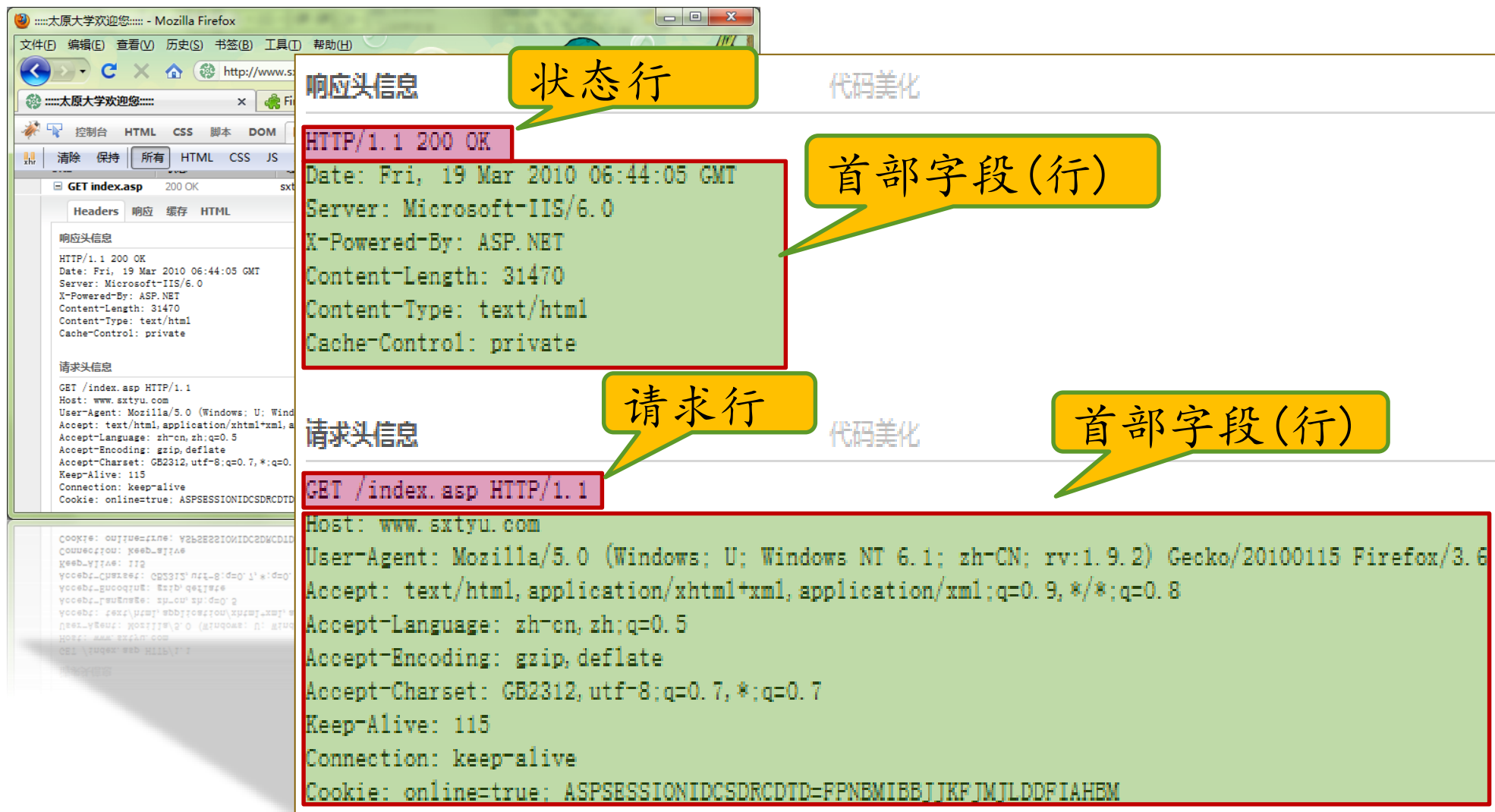
HTTP报文结构--首部字段或消息头

头(header)	类型	说明
User-Agent	请求	关于浏览器和它平台的信息，如Mozilla5.0
Accept	请求	客户能处理的页面的类型，如text/html
Accept-Charset	请求	客户可以接受的字符集，如Unicode-1-1
Accept-Encoding	请求	客户能处理的页面编码方法，如gzip
Accept-Language	请求	客户能处理的自然语言，如en(英语)，zh-cn(简体中文)
Host	请求	服务器的DNS名称。从URL中提取出来，必需。
Authorization	请求	客户的信息凭据列表
Cookie	请求	将以前设置的Cookie送回服务器，可用来作为会话信息

HTTP报文结构--首部字段或消息头

头(header)	类型	说明
Date	双向	消息被发送时的日期和时间
Server	响应	关于服务器的信息，如Microsoft-IIS/6.0
Content-Encoding	响应	内容是如何被编码的（如gzip）
Content-Language	响应	页面所使用的自然语言
Content-Length	响应	以字节计算的页面长度
Content-Type	响应	页面的MIME类型
Last-Modified	响应	页面最后被修改的时间和日期，在页面缓存机制中意义重大
Location	响应	指示客户将请求发送给别处，即重定向到另一个URL
Set-Cookie	响应	服务器希望客户保存一个Cookie

HTTP报文结构—实例



The image shows a screenshot of a Mozilla Firefox browser window displaying the HTTP response headers for a GET request to `http://www.sxtyu.com/index.asp`. The browser's developer tools (F12) are open, showing the 'Headers' tab. The response status is `200 OK`. The response headers include `Date`, `Server`, `X-Powered-By`, `Content-Length`, `Content-Type`, and `Cache-Control`. The request headers include `Host`, `User-Agent`, `Accept`, `Accept-Language`, `Accept-Encoding`, `Accept-Charset`, `Keep-Alive`, `Connection`, and `Cookie`.

Response Headers (响应头信息):

```
HTTP/1.1 200 OK
Date: Fri, 19 Mar 2010 06:44:05 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
Content-Length: 31470
Content-Type: text/html
Cache-Control: private
```

Request Headers (请求头信息):

```
GET /index.asp HTTP/1.1
Host: www.sxtyu.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; zh-CN; rv:1.9.2) Gecko/20100115 Firefox/3.6
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-cn,zh;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: GB2312,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Cookie: online=true; ASPSESSIONIDCSDRCDDT=FPNEMIBBJJKFJWJLDDFIAHEM
```

Annotations:

- 状态行 (Status Line):** `HTTP/1.1 200 OK`
- 首部字段 (行) (Header Fields (Line)):** `Date: Fri, 19 Mar 2010 06:44:05 GMT`
- 请求行 (Request Line):** `GET /index.asp HTTP/1.1`
- 首部字段 (行) (Header Fields (Line)):** `Host: www.sxtyu.com`

HTTP协议--Cookie

- **Cookie:**当我们使用自己的电脑通过浏览器进行访问网页的时候，服务器就会生成一个证书并返回给我的浏览器并写入我们的本地电脑。这个证书就是cookie
- **Set-Cookie:** PHPSESSID=l89hpkrks06k82juj4vjoc4mu6; path=/
32位长度16进制编码
- **C:\Windows\TEMP**

HTTP协议与会话管理

Cookie



Name	Value	Domain	Path	Expires / ...	Size	HTTP	Secure
__cfduid	7d493e0a2422287207df0ac9c8f21423187255	.gravatar...	/	2016-02...	51	✓	
_ga	1341763833.1423474362	.github.c...	/	2017-03...	30		
_gh_sess	ONzaW9uX2lkajYmY0NjYzZGU3ZTc2MzhkM2Rk...	.github.c...	/	Session	226	✓	✓
_octo	399677830.1423474363	.github.c...	/	2017-02...	31		
doccom_user	9	.github.c...	/	2035-02...	18	✓	✓
logged_in		.github.c...	/	2035-02...	12	✓	✓
tz	2FShanghai	github.c...	/	Session	17		✓
user_session	bNn58QGr-dPPnczs28x5FtDPwHbyzVloOTz69n0MV...	github.c...	/	2015-04...	92	✓	✓

字段	说明
Name	Cookie名称
Value	Cookie的值
Domain	用于指定Cookie的有效域
Path	用于指定Cookie的有效URL路径
Expires	用于设定Cookie的有效时间
Secure	如果设置该属性，仅在HTTPS请求中提交Cookie
Http	其实应该是HttpOnly，如果设置该属性，客户端JavaScript无法获取Cookie值

HTTP协议--Session

- Session机制是一种服务器端的机制。
- 当程序需要为某个客户端的请求创建一个session的时候，服务器首先检查这个客户端的请求里是否已包含了一个session标识 - 称为 **session id**，如果已包含一个session id则说明以前已经为此客户端创建过session，服务器就按照session id把这个 session检索出来使用（如果检索不到，可能会新建一个），如果客户端请求不包含session id，则为此客户端创建一个session并且生成一个与此session相关联的session id，session id的值应该是一个既不会重复，又不容易被找到规律以仿造的字符串，这个session id将被在本次响应中返回给客户端保存

Session

- Session 是 用于保持状态的基于 Web服务器的方法
- 允许通过将对象存储在 Web服务器的内存中在整个用户会话过程中保持任何对象

Session 与 Cookie 的区别

- Cookie的数据保存在客户端浏览器，Session保存在服务器
- 服务端保存状态机制需要在客户端做标记，所以Session可能借助Cookie机制
- Cookie通常用于客户端保存用户的登录状态

抓包分析能力

■ 抓包工具：

- 浏览器自带
- **Fiddler**
- **HttpWatch**
- **Charles**



Question
