简介：

黑夜小怪，在武汉工作，QQ：244240753

个人博客：http://www.cnblogs.com/heiyexiaoguai/

擅长领域：持续集成、接口、小工具开发（java、C#、shell、python）、技术分享

QQ群：283440449

长按识别二维码

关注「 飞测 」

说说我正在经历和即将经历的测试百态

支付宝扫一扫，向我付款

HTTP协议分享

1、 • 初识HTTP

2、 • HTTP的工作原理

3、 • 请求行、状态行和消息报头

4、 • 缓存和认证

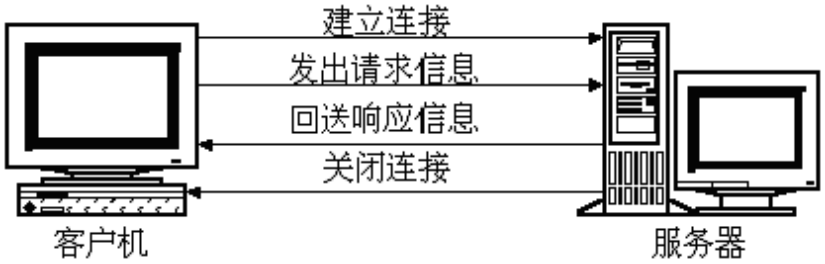
5、 • Fiddler的基本介绍

初识HTTP协议

HTTP的介绍：

- 英文名： Hypertext Transfer Protocol
- **中文名：** 超文本传输协议
- **方式：** IP+端口，默认HTTP的端口号为80，HTTPS的端口号为443。
- **功能：** 在服务器和客户机之间传输超文本文件
- **模式：** 请求响应模式
- **版本：**
  1、HTTP/1.0:非持续连接；
  2、HTTP/1.1:目前主要使用的，是持续连接；
  3、HTTP-NG ：

HTTP 的原理



HTTP 1.0

1、连接过程是短暂的

2、每次连接只处理一个请求和响。

3、每一个页面的访问，浏览器与WEB服务器都要建立一次单独的连接。

4、所有通讯都是完全独立分开的请求和响应对

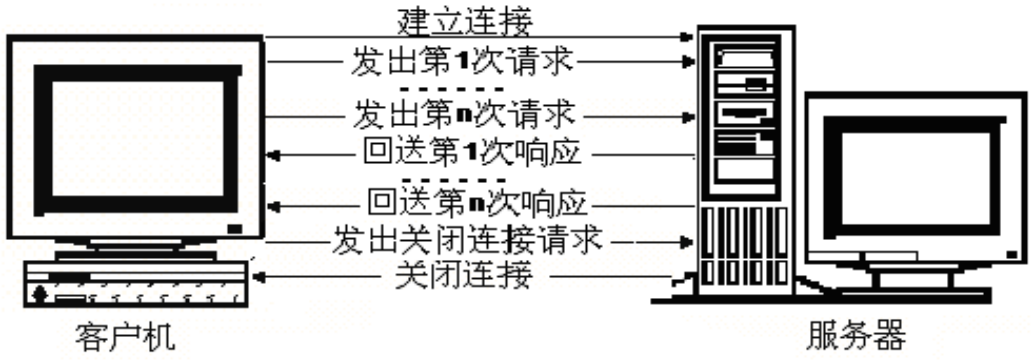
HTTP 1.1

1、在一个TCP连接上可以传送多个HTTP请求和响应

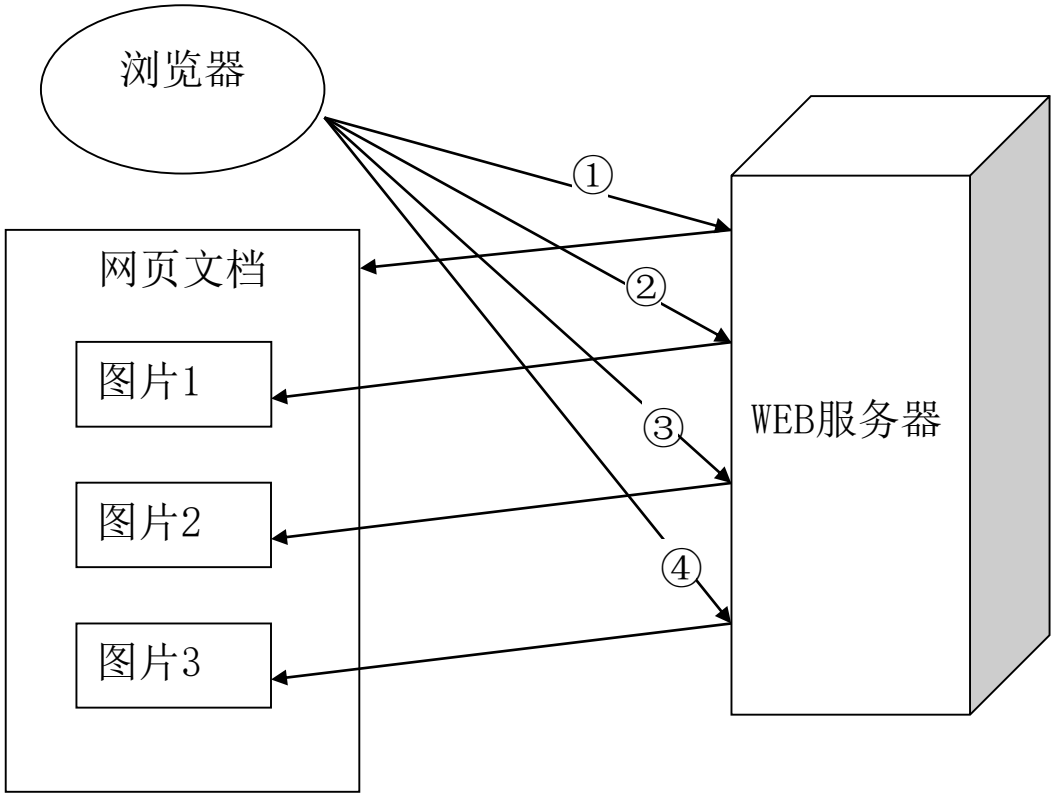
2、多个请求和响应过程可以重叠进行

3、增加了更多的请求头和响应头

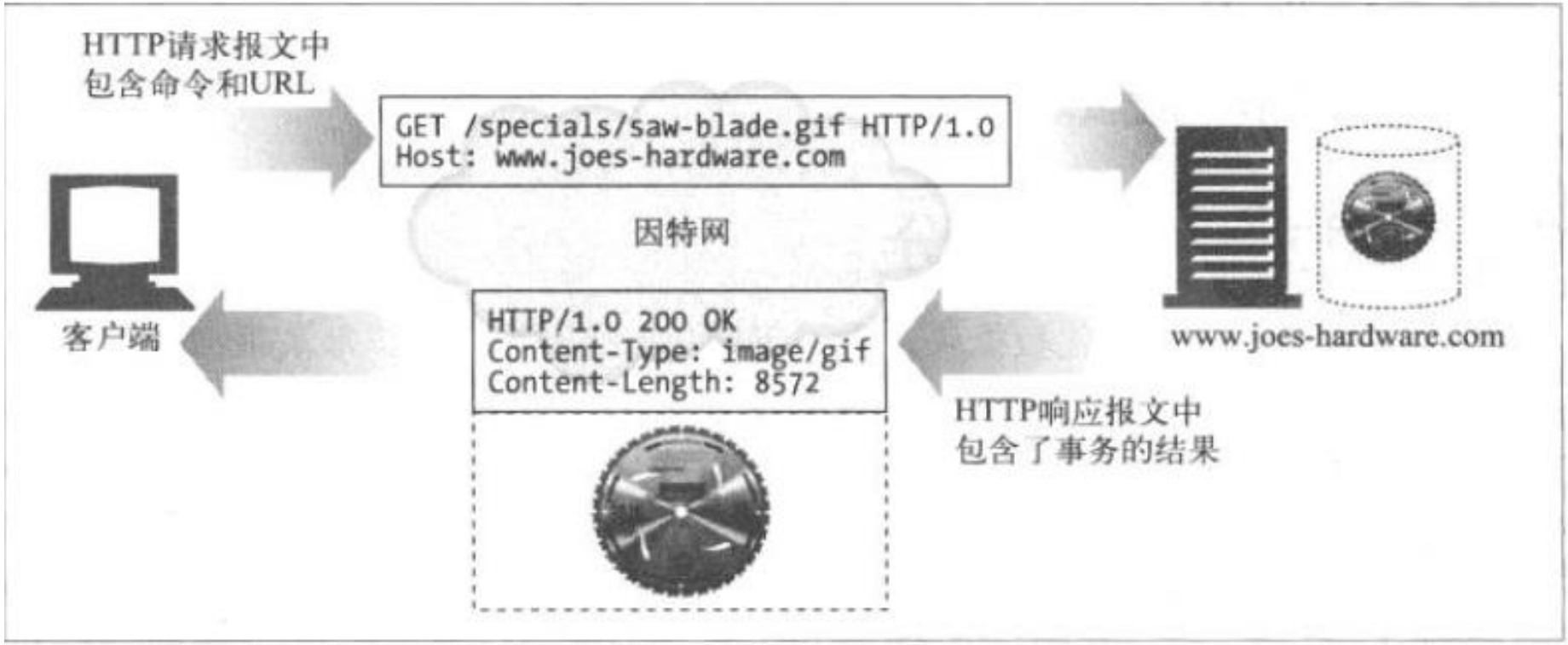
4、Connection报头来控制



浏览器访问多图网页的过程



简单的例子：后用IE浏览器请求页面 ：



请求消息：

GET /simple.htm HTTP/1.1

Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, */*

Accept-Language: zh-cn

Accept-Encoding: gzip, deflate

User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; .NET CLR 2.0.50727)

Host: localhost:8080

Connection: Keep-Alive

请求包

响应消息：

HTTP/1.1 200 OK

Server: Microsoft-IIS/5.1

X-Powered-By: ASP.NET

Date: Fri, 03 Mar 2006 06:34:03 GMT

Content-Type: text/html >

Accept-Ranges: bytes

Last-Modified: Fri, 03 Mar 2006 06:33:18 GMT

ETag: "5ca4f75b8c3ec61:9ee"

Content-Length: 37

<CR>

<html><body>hello world</body></html>

响应包

HTTP协议分享

1、 • 初识HTTP

2、 • HTTP的工作原理

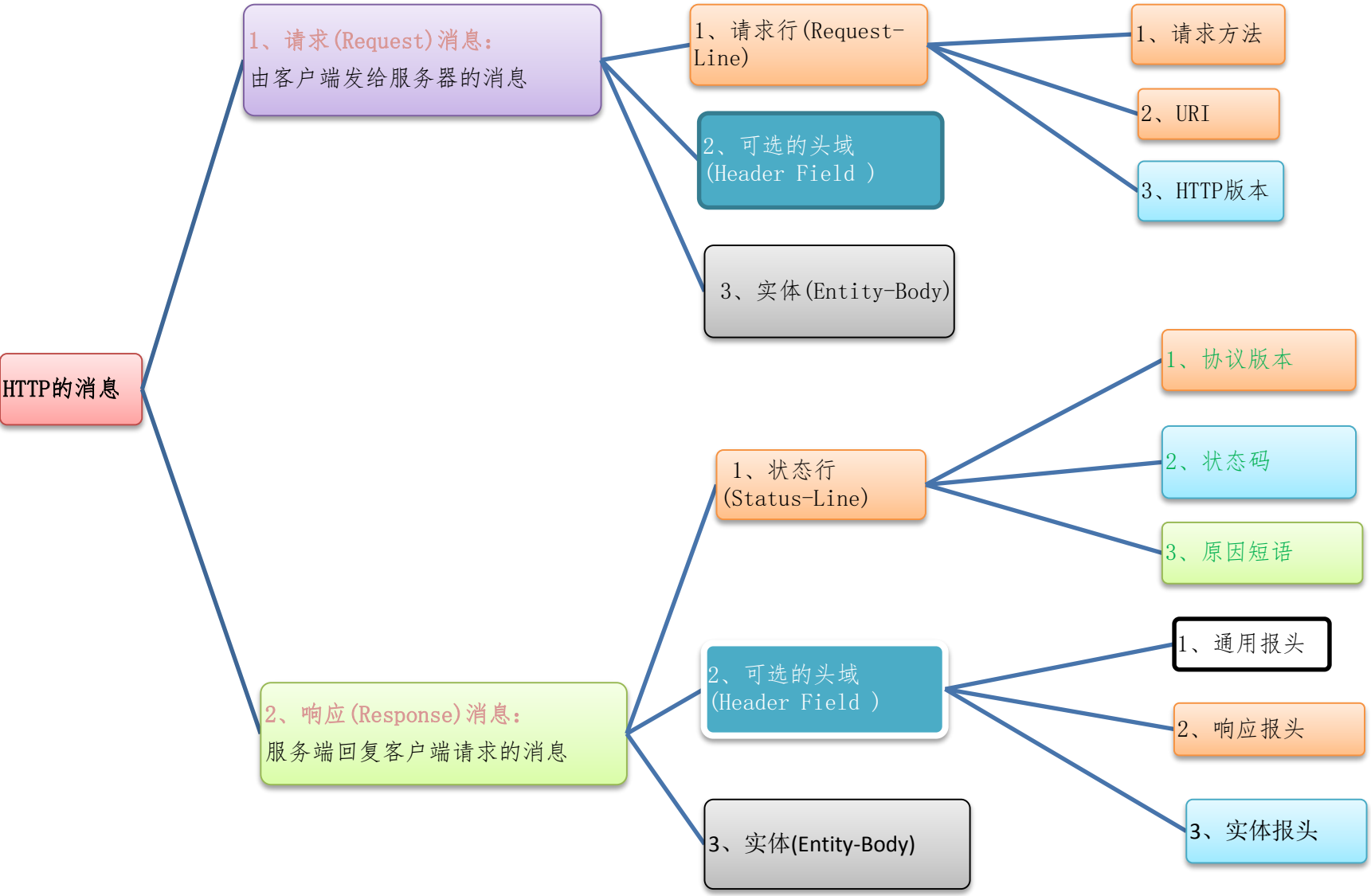
3、 • 请求行、状态行和消息报头

4、 • 缓存和认证

5、 • Fiddler的基本介绍

HTTP的消息类型

HTTP的消息类型分类：



HTTP协议分享

1、 • 初识HTTP

2、 • HTTP的工作原理

3、 • 请求行、状态行和消息报头

4、 • 缓存和认证

5、 • Fiddler的基本介绍

HTTP请求消息

请求消息的结构：

```
GET /books/java.html HTTP/1.1
Accept: */*
Accept-Language: en-us
Connection: Keep-Alive
Host: localhost
Content-Length: 0
User-Agent: Mozilla/4.0
Accept-Encoding: gzip, deflate
```

←请求行：
① 请求方法
② URI（被请求资源所处的地址
③ HTTP版本

←多个消息头：

←一个空行

HTTP响应消息

响应消息的结构：

- 举例：

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Thu, 13 Jul 2000 05:46:53 GMT
Content-Length: 2291
Content-Type: text/html
Cache-control: private

<HTML>
<BODY>
……
```

←状态行：
① 协议版本
② 状态码
③ 原因短语

←多个消息头

←一个空行

←实体内容

HTTP请求和响应消息

请求消息：		响应消息：
POST /simple.htm HTTP/1.1	起始行	HTTP/1.1 200 OK
Accept: image/gif, */* Accept-Language: zh-cn Accept-Encoding: gzip, deflate Host: localhost:8080 Connection: Keep-Alive	消息头	Server: Microsoft-IIS/5.1 X-Powered-By: ASP.NET Date: Fri, 03 Mar 2006 06:34:03 GMT Content-Type: text/html > Accept-Ranges: bytes Last-Modified: Fri, 03 Mar 2006 06:33:18 GMT ETag: "5ca4f75b8c3ec61:9ee" Content-Length: 37
name=myname&pwd=mypwd	实体内容	<html><body>hello world</body></html>

请求方法

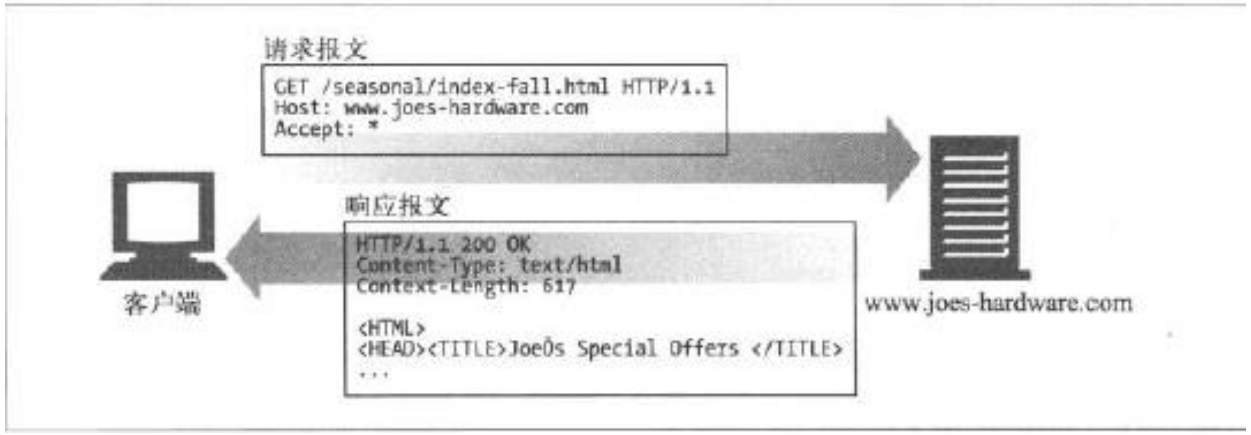
请求方法种类：

方法名	备注	是否包含实体内容
GET	从服务器获取一份文档	否
HEAD	只从服务器获取文档的消息头	否
POST	向服务器发送要处理的数据	是
PUT	将请求的实体内容存储在服务器上	是
DELETE	请求源服务器删除Request-URI标识的资源	否
TRACE	对可能经过代理服务器传送到服务器上去的报文进行跟踪	否
OPTIONS	决定可以在服务器上执行哪些方法	否

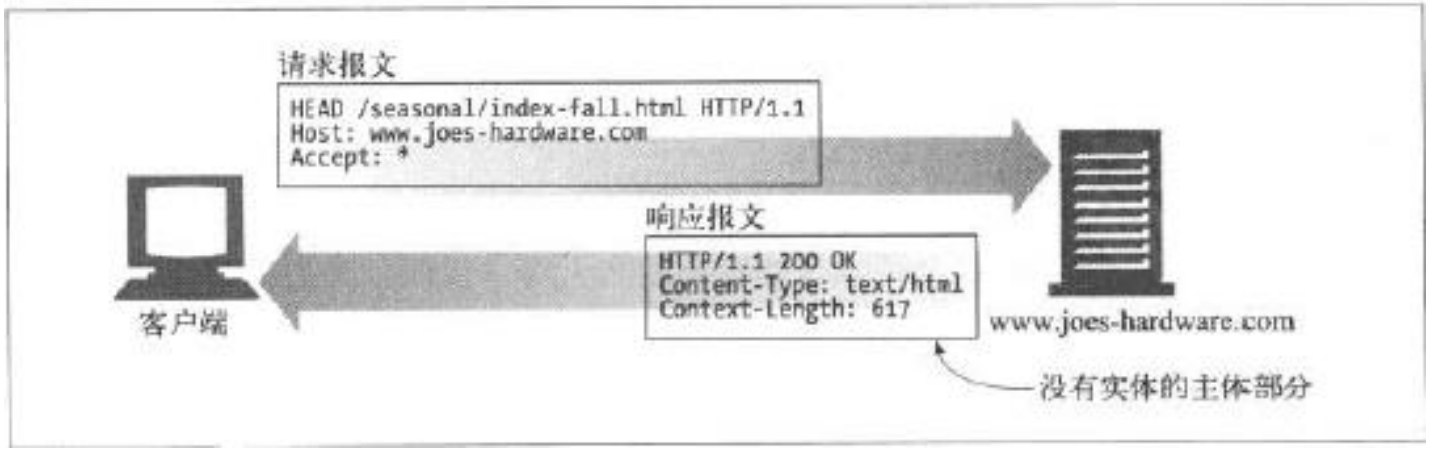
GET方式：

举例： *GET /servlet/ParamsServlet? param1=abc&param2=xyz HTTP/1.1*

特点： *传送的数据量是有限制的，一般限制在1KB以下*



HEAD方法：

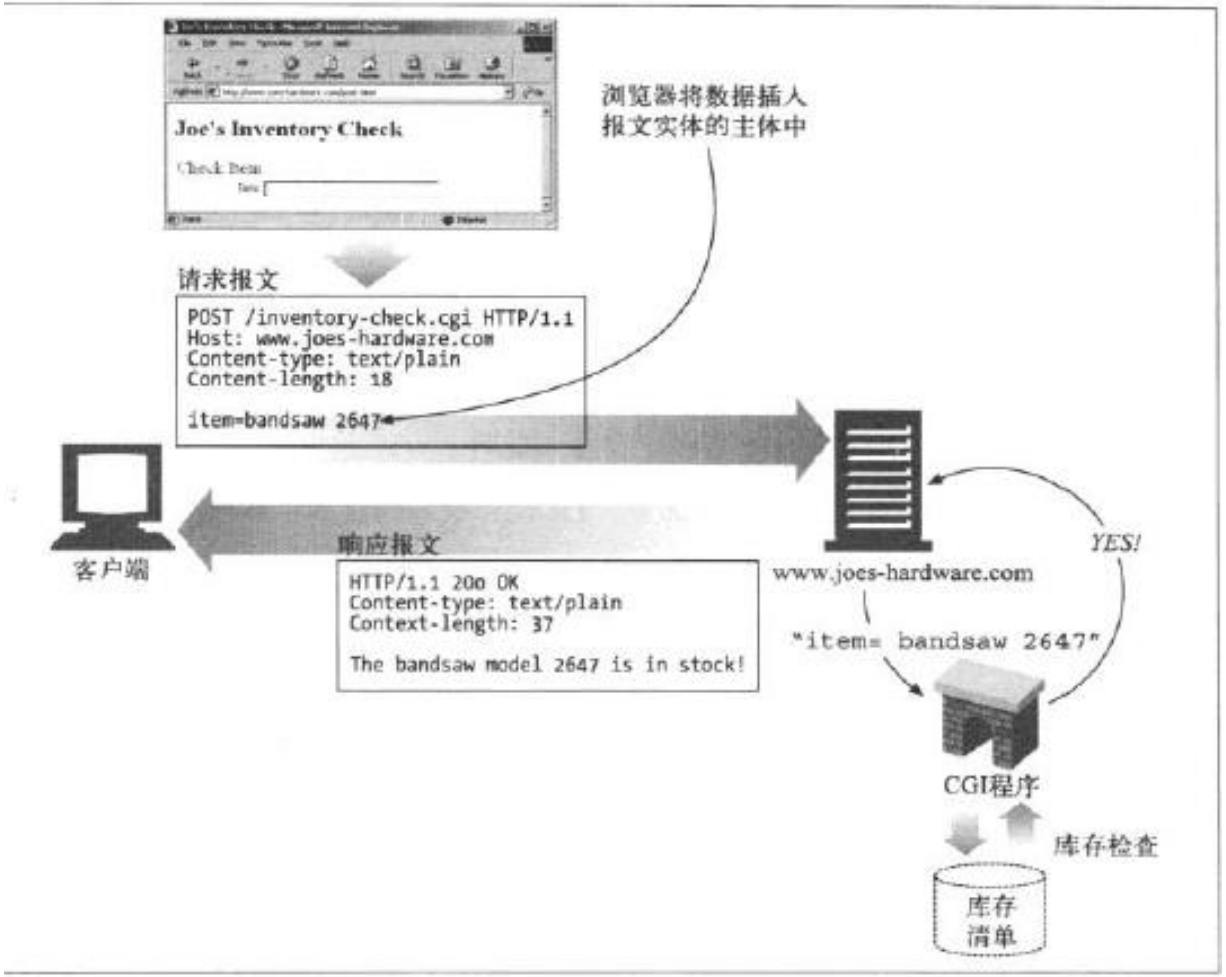


举例：

```
POST /servlet/ParamsServlet HTTP/1.1
Host:
Content-Type: application/x-www-form-urlencoded
Content-Length: 28

param1=abc&param2=xyz
```

特点：*传送的数据量要比GET方式传送的数据量大得多 。*



响应状态码

响应状态码可类别：

状态码	分类	整体范围	已定义范围
1XX	信息提示	100~199	100~101
2XX	成功	200~299	200~206
3XX	重定向	300~399	300~305
4XX	客户端错误	400~499	400~415
5XX	服务端错误	500~599	500~505

响应状态码

- **200（正常）**

  表示一切正常，返回的是正常请求结果。

- **206（部分内容）**

  客户发送了一个带有Range头（要求服务器只返回文档中的部分内容）的GET请求，服务器按要求完成了这个请求。

- **302/307（临时重定向）**

  指出被请求的文档已被临时移动到别处，此文档的新的URL在Location响应头中给出。

- **304（未修改）**

  表示客户机缓存的版本是最新的，客户机应该继续使用它。

- **401/407（未经授权）**

  表示客户机访问的是一个受口令和密码保护的页面，结合使用一个WWW-Authenticate响应头提示客户机应重新发出一个带有Authorization头的请求消息。

- **404（找不到）**

  服务器上不存在客户机所请求的资源

- **500（内部服务器错误）**

  服务器端的CGI、ASP、JSP等程序发生错误。

HTTP协议分享

1、 • 初识HTTP

2、 • HTTP的工作原理

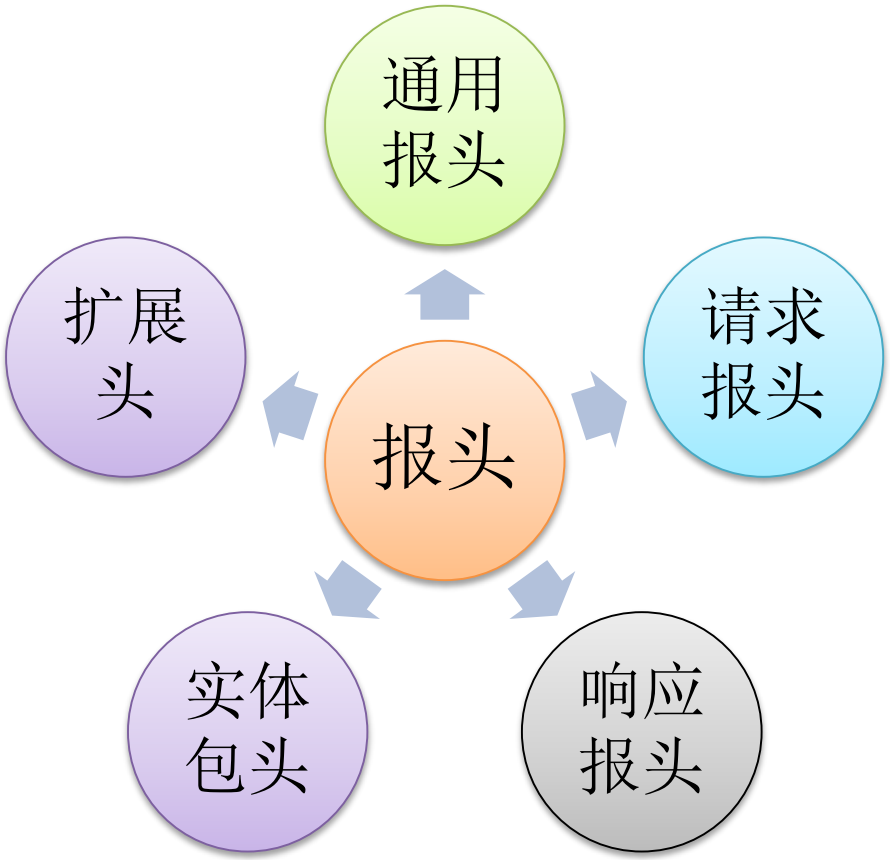
3、 • 请求行、状态行和消息报头

4、 • 缓存和认证

5、 • Fiddler的基本介绍

消息头（报头）

分类：



特点：

1、消息头字段名是不区分大小写的，但习惯上将每个单词的第一个字母大写。

2、各行消息头无顺序排列。

3、许多请求头字段都允许多值，以逗号分隔。举例：Accept-Encoding: gzip, compress

4、有些头字段可以出现多次，例如，响应消息中可以包含有多个“Warning”头字段。

通用信息头

- **通用信息头**字段既能用于请求消息，也能用于响应消息，它包括一些与被传输的实体内容没有关系的常用消息头字段。
  - ✓ **Cache-Control**: no-cache （*）
  - ✓ **Connection**: close/Keep-Alive （*）
  - ✓ **Date**: Tue, 11 Jul 2000 18:23:51 GMT
  - ✓ **Pragma**: no-cache （*）
  - ✓ **Trailer**: Date
  - ✓ **Transfer-Encoding**: chunked （*）
  - ✓ **Upgrade**: HTTP/2.0, SHTTP/1.3
  - ✓ **Via**: HTTP/1.1 Proxy1, HTTP/1.1 Proxy2
  - ✓ **Warning**: any text

请求头

- **请求头字段**用于客户端在请求消息中向服务器传递附加信息，主要包括客户端可以接受的数据类型、压缩方法、语言、以及发出请求的超链接所属网页的URL地址等信息。

- ✓ **Accept**: text/html,image/* （*）
- ✓ **Accept-Charset**: ISO-8859-1,unicode-1-1 （*）
- ✓ **Accept-Encoding**: gzip,compress （*）
- ✓ **Accept-Language**: en-gb,zh-cn （*）
- ✓ **Authorization**: Basic enh4OjEyMzQ1Ng== （*）
- ✓ **Expect**: 100-continue
- ✓ **From**: zxx@it315.org
- ✓ **Host**: www.it315.org:80 （*）
- ✓ **If-Match**: "xyzzy", "r2d2xxxx"
- ✓ **If-Modified-Since**: Tue, 11 Jul 2000 18:23:51 GMT （*）
- ✓ **If-None-Match:** "xyzzy", "r2d2xxxx"
- ✓ **If-Range**: Tue, 11 Jul 2000 18:23:51 GMT
- ✓ **If-Unmodified-Since**: Tue, 11 Jul 2000 18:23:51 GMT
- ✓ **Max-Forwards**: 1
- ✓ **Proxy-Authorization**: Basic enh4OjEyMzQ1Ng==
- ✓ **Range**: bytes=100-599 （*）
  Range: bytes=100-
  Range: bytes=-100
- ✓ **Referer**: http://www.it315.org/index.jsp （*）
- ✓ **TE**: trailers,deflate
- ✓ **User-Agent**: Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 5.0)

响应头

- 响应头字段用于服务器在响应消息中向客户端传递附加信息，包括服务程序名，被请求资源需要的认证方式，被请求资源已移动到的新地址等信息。

  - ✓ **Accept-Range**: bytes （*）
  - ✓ **Age**: 315315315
  - ✓ **Etag**: b38b9-17dd-367c5dcd
  - ✓ **Location**: http://www.it315.org/index.jsp （*）
  - ✓ **Proxy-Authenticate**: BASIC realm="it315"
  - ✓ **Retry-After**: Tue, 11 Jul 2000 18:23:51 GMT
  - ✓ **Server**: Microsoft-IIS/5.0 （*）
  - ✓ **Vary**: Accept-Language
  - ✓ **WWW-Authenticate**: BASIC realm="it315" （*）

实体头

- 实体头用作实体内容的元信息，描述了实体内容的属性，包括实体信息类型、长度、压缩方法、最后一次修改时间、数据有效期等。
  - ✓ **Allow**: GET,POST
  - ✓ **Content-Encoding**: gzip （*）
  - ✓ **Content-Language**: zh-cn （*）
  - ✓ **Content-Length**: 80 （*）
  - ✓ **Content-Location**: http://www.it315.org/java_cn.html
  - ✓ **Content-MD5**: ABCDABCDABCDABCDABCDAB==
  - ✓ **Content-Range**: bytes 2543-4532/7898 （*）
  - ✓ **Content-Type**: text/html; charset=GB2312 （*）
  - ✓ **Expires**: Tue, 11 Jul 2000 18:23:51 GMT （*）
  - ✓ **Last-Modified**: Tue, 11 Jul 2000 18:23:51 GMT （*）

扩展头

- 在HTTP消息中，也可以使用一些在HTTP 1.1正式规范里没有定义的头字段，这些头字段统称为自定义的HTTP头或扩展头，它们通常被当作是一种实体头处理。
- 几个常用的扩展头字段:
  - Cookie、
  - Set-Cookie、
  - Refresh
  - Content-Disposition

- **Refresh头字段**
  - ✓ Refresh: 1
  - ✓ Refresh: 1;url=http://www.it315.org
- **Content-Disposition头字段**

  Content-Type: application/octet-stream

  Content-Disposition: attachment; filename=aaa.zip 表示下载该文件，文件名是aaa.zip

常用报头详细介绍

1、Cache-Control普通报头域指定请求和响应遵循的缓存机制

请求时：no-cache、no-store、max-age、max-stale、min-fresh、only-if-cached

响应时：public、private、no-cache、no-store、no-transform、must-revalidate、proxy-revalidate、max-age。

2、Pragma：指定“no-cache”值表示服务器必须返回一个刷新后的文档，即使它是代理服务器而且已经有了页面的本地拷贝；

- **打开新窗口**

  如果指定cache-control的值为private、no-cache、must-revalidate，那么打开新窗口访问时都会重新访问服务器。而如果指定了max-age值，那么在此值内的时间里就不会重新访问服务器

  例如：Cache-control: max-age=5

  表示当访问此网页后的5秒内再次访问不会去服务器

- **在地址栏回车**

  如果值为private或must-revalidate（和网上说的不一样），则只有第一次访问时会访问服务器，以后就不再访问。如果值为no-cache，那么每次都会访问。如果值为max-age，则在过期之前不会重复访问。

- **按后退按扭**

  如果值为private、must-revalidate、max-age，则不会重访问，而如果为no-cache，则每次都重复访问

- **按刷新按扭**

  无论为何值，都会重复访问

常用报头详细介绍

3、Date：普通报头域表示消息产生的日期和时间

4、Connection普通报头域表示是否需要持久连接。值为“Keep-Alive”或”close”, HTTP1.1默认是持久连接

5、Expect: 100-continue： Server看到之后，如果回100 (Continue) 这个状态代码，客户端就继续发request body

6、Host：初始URL中的主机和端口，HTTP1.1新加的头（必须）

7、Accept请求报头域用于指定客户端接受哪些类型的信息

8、Accept-Charset：Charset:iso-8859-1,gb2312.如果在请求消息中没有设置这个域，缺省是任何字符集都可以接受。

常用报头详细介绍

9、Accept-Encoding：用于指定可接受的内容编码，比如gzip， deflate 。Servlet能够向支持gzip的浏览器返回经gzip编码的HTML页面。许多情形下这可以减少5到10倍的下载时间；

10、Accept-Language：浏览器所希望的语言种类，当服务器能够提供一种以上的语言版本时要用到，例如zh-cn

11、Authorization请求报头域主要用于证明客户端有权查看某个资源，通常出现在对服务器发送的WWW-Authenticate头的应答中；

12、Cookie：这是最重要的请求头信息之一；

13、**If-Modified-Since：**客户端存取的该资源最后一次修改的时间，同Last-Modified，只有当所请求的内容在指定的日期之后又经过修改才返回它，否则返回304“Not Modified”应答；

14、If-None-Match：客户端存取的该资源的检验值，同ETag。

15、Referer：包含一个URL，用户从该URL代表的页面出发访问当前请求的页面

常用报头详细介绍

15、User-Agent：请求报头域允许客户端将它的操作系统、浏览器和其它属性告诉服务器，IE： Mozilla/4.0(compatible;MSIE6.0;Windows NT 5.0)

16、Location响应报头域用于重定向接受者到一个新的位置（常用在更换域名的时候）

17、Content-Encoding：文档的编码（Encode）方法，例如：gzip，见“2.5响应头”；

18、**Content-Language**实体报头域描述资源所用的自然语言，例如：zh-cn；

19、**Content-Length**：表示内容长度，eg：80，只有当浏览器使用持久HTTP连接时才需要这个数据；

20、**Content-Location**：表示客户应当到哪里去提取文档，例如：http://www.dfdf.org/dfdf.html，可参考“2.5响应头”；

常用报头详细介绍

21、Content-MD5： MD5 实体的一种MD5摘要，用作校验和。发送方和接受方都计算MD5摘要，接受方将其计算的值与此头标中传递的值进行比较。Eg1：Content-MD5: <base64 of 128 MD5 digest>。

22、Content-Range： 随部分实体一同发送；标明被插入字节的低位与高位字节偏移，也标明此实体的总长度。Eg1：Content-Range: 1001-2000/5000，eg2：bytes 2543-4532/7898

23、Content-Type： 标明发送或者接收的实体的MIME类型。Eg：text/html; charset=GB2312　　主类型/子类型；

24、Expires： 指明应该在什么时候认为文档已经过期，从而不再缓存它。为0证明不缓存；

25、Last-Modified： WEB 服务器认为对象的最后修改时间，比如文件的最后修改时间，动态页面的最后产生时间等等。 只有改动时间迟于指定时间的文档才会返回，否则返回一个304（Not Modified）状态,例如：Last-Modified：Tue, 06 May 2008 02:42:43 GMT.

常用报头详细介绍

26、Set-Cookie： 服务器设置cookie

27、Refresh： 1;url=http://www.dfdf.org //过1秒跳转到指定位置；

28、Content-Disposition： 头字段,可参考“2.5响应头”；

29、Content-Type： WEB 服务器告诉浏览器自己响应的对象的类型。

- eg1：Content-Type：application/xml ；
- eg2：applicaiton/octet-stream；

30、Content-Disposition： attachment; filename=aaa.zip。

31、 **WWW-Authenticate** 响应报头域必须被包含在401（未授权的）响应消息中，客户端收到401 响应消息时候，并发送Authorization 报头域请求服务器对其进行验证时，服务端响应报头就包含该报头域。eg：WWW-Authenticate:Basic realm="Basic Auth Test!" //可以看出服务器对请求资源采用的是基本验证机制。

HTTP协议分享

1、 • 初识HTTP

2、 • HTTP的工作原理

3、 • 请求行、状态行和消息报头

4、 • 缓存和认证

5、 • Fiddler的基本介绍

缓存（Cache　）

- 服务器在响应消息中用Set-Cookie头将Cookie的内容回送给客户端，客户端在新的请求中将相同的内容携带在Cookie头中发送给服务器。从而实现会话的保持。
- 流程如下图所示：

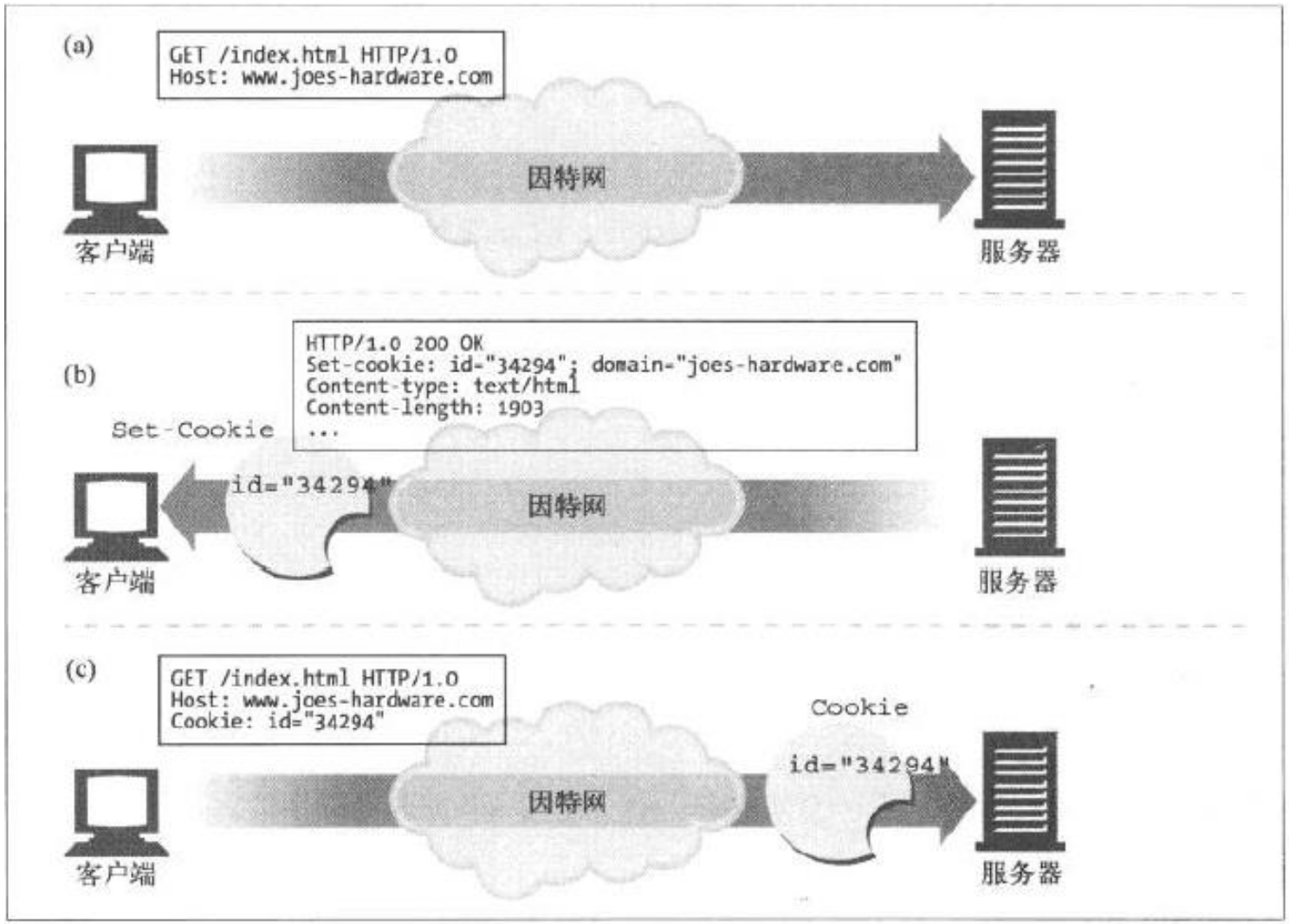
缓存（Cache　）

- 服务器收到请求时，会在200 OK中回送该资源的Last-Modified和ETag头，客户端将该资源保存在cache中，并记录这两个属性。当客户端需要发送相同的请求时，会在请求中携带If-Modified-Since和If-None-Match两个头。两个头的值分别是响应中Last-Modified和ETag头的值。未发生变化，客户端不需要重新下载，返回304响应
- 流程如下图所示：

缓存（Cache　）

常见流程如下图所示：

- 当COOKIE过期后浏览器就不会再发送它了。



缓存（Cache　）

与Cookie相关的HTTP扩展头

1. **Cookie**：客户端将服务器设置的Cookie返回到服务器；
2. **Cookie2 (RFC2965)）**：客户端指示服务器支持Cookie的版本；
3. **Cache-Control**：更细致的控制缓存的内容
4. **Set-Cookie**：服务器向客户端设置Cookie；
5. **Set-Cookie2 (RFC2965)**：服务器向客户端设置Cookie。
6. **Expires**：指示响应内容过期的时间，格林威治时间GMT
7. **Date**：服务器的时间
8. **If-Modified-Since**：客户端存取的该资源最后一次修改的时间，同Last-Modified。
9. **If-None-Match**：客户端存取的该资源的检验值，同ETag。
10. **Last-Modified**：响应中资源最后一次修改的时间
11. **ETag**：响应中资源的校验值，在服务器上某个时段是唯一标识的。

简介：

黑夜小怪，在武汉工作，QQ：244240753

个人博客：http://www.cnblogs.com/heiyexiaoguai/

擅长领域：持续集成、接口、小工具开发（java、C#、shell、python）、技术分享

QQ群：283440449

长按识别二维码

关注「 飞测 」

说说我正在经历和即将经历的测试百态

支付宝扫一扫，向我付款