# HTTP

## 简介

HTTP（Hyper Text Transformer Protocol，超文本传输协议）是一种通信协议，它允许将超文本标记语言（HTML）文档从Web服务器传送到客户端的浏览器。

它是一个应用层协议，承载于TCP协议之上

由请求和响应构成，是一个标准的客户端服务器模型

## 版本

作为Web文档传输协议的HTTP，版本更新十分缓慢，目前只更新了三个版本：

HTTP/0.9

HTTP/1.0

HTTP/1.1

## 应用

HTTP是整个Web基础，许多应用都离不开对HTTP协议的认识：

WebService=HTTP+XML

Reset=HTTP+JSON

API的实现：HTTP+XML/JSON

采集、小偷站

QQ、迅雷等桌面应用软件

# TCP/IP协议

## 分层管理

TCP/IP协议族最重要的一点就是分层。按层次分别分为以下4层：

应用层：负责处理特定的应用程序，TCP/IP协议族内预存了各类通用的应用服务。比如FTP、DNS等等，HTTP也在该层。

传输层：对上层应用层提供处于网络连接中两台计算机之间的数据传输。该层有两个不同的协议：TCP和UDP。

网络层：处理网络上流动的数据包，该层协议有：IP、ICMP、IGMP。

数据链路层：处理连接网络的硬件部分。包括操作系统、硬件设备驱动、网卡、光纤等。

## 流程图



# IP/TCP/DNS

## IP协议

### 简介

IP（Internet Protocol），翻译成网络协议，位于网络层。

### 作用

IP的作用：把各种数据包传送给对方。如果要保证正确传送到对方那里，则需要满足各类条件，其中最重要的两个条件时IP地址和MAC地址。

IP地址：指明了节点被分配到的地址

MAC地址：是指网卡所属的固定地址

区别：IP地址可变换，但MAC地址基本上不会更改。

### ARP

使用ARP协议凭借MAC地址进行通信：

ARP是一种解析地址的协议，根据通信放的IP地址就可以反向查出对应的MAC地址。



## TCP协议

### 简介

TCP（Transmission Control Protocol），翻译成传输控制协议，位于传输层。

### 作用

提供可靠的字节流服务。

### 三次握手

TCP采用三次握手的策略保证提供可靠的服务。



## DNS

### 简介

DNS（Domain Name System），翻译成域名系统，位于应用层。

### 作用

提供域名到IP地址之间的解析服务。



## 联系



# URI与URL

## URI

### 简介

URI（Uniform Resource Identifier），翻译为统一资源**标识符**，是一个用于标识某一互联网资源名称的字符串。

### URI格式

格式：

<http://user:pass@www.example.com:80/home/index.html?age=11#mask>

http：协议方案名

user:pass：登录信息（认证）

[www.example.com](http://www.example.com)：服务器地址

80：端口号

/home/index.html：文件路径

age=11：查询字符串

mask：片段标识符

协议方案名：http:、https:、ftp:等，在获取资源时要指定协议类型

登录信息（认证）：指定用户名和密码作为从服务器端获取资源时必要的登录信息，此项是可选的。

不带用户名和密码访问FTP服务器：<ftp://test.com/>

只带用户名访问FTP服务器：<ftp://username@baidu.com>

带用户名和密码访问FTP服务器：ftp://username:password@baidu.com/

服务器地址：使用绝对URI必须指定待访问的服务器地址。

服务器端口号：指定服务器连接的网络端口号，此选项是可选的。常用默认端口：HTTP（80）、FTP（21）、Telnet（23）、HTTPS（443）、SMTP（25）、POP3（110）等等。

路径：指定服务器上的文件路径来定位特定资源。格式为：/home/index.html

参数：为应用程序提供访问资源所需的附加信息，例如：ftp:/127.27.27.27/pub/pic;type=d

查询字符串：针对已指定的文件路径内的资源，可以使用查询字符串传入任意参数，即传查询条件到服务器，来缩小请求资源的范围。此项是可选的。



片段标识符：通常可标记出以获取资源中的子资源（文档内的某一位置），此选项是可选的。

例如：http://www.baidu.com/path/php/#footer

## URL

### 简介

URL（Uniform Resource Location），翻译为统一资源**定位符**，它描述一台特定服务器上某特定资源的特定位置。

URI用字符串标识某一互联网资源，而URL表示资源的地点（资源所处的位置）。由此可见，URL是URI的子集。

URI由两个主要的子集URL（通过位置识别资源）和URN（通过名字识别资源）构成。

注：URN（Uniform Resource Name，统一资源**名称**）没有得到广泛应用。

### 组件作用

大多数URL语法都建立在这个由9部分构成的通用格式上：



### 相对URL

相对URL：URL是不完整的

基础URL：必须要有一个参考点



#### 解析过程

1、找到基础URL



2、将相对URL转为绝对URL



#### 自动拓展URL

自动拓展URL的两种方式：

1. 主机名扩展

例如：输入baidu🡪浏览器自动拓展为www.baidu.com

1. 历史扩展

例如：输入bai🡪浏览器自动填充baidu.com

### 编码机制

用ASCII字符集中的有限子集对任意ASCII字符集外的字符值或数据进行编码。

格式：一个百分号（%）、后面跟两个表示字符ASCII码的十六进制数



### URL保留及受限字符

所谓保留字就是那些在URL中具有特定意义的字符。

受限（不安全）字符是指那些在URL中没有特殊含义，但在URL所在的上下文中可能具有特殊意义的字符，例如双引号。



# HTTP请求方法

## GET

### WampServer

Wamp就是Windows Apache MySQL PHP集成安装环境，即在windows下的apache、php和mysql的服务器软件。

### GET请求方法

## POST

### POST请求方法

POST请求方法的作用：向服务器发送数据



### 区别

POST和GET请求方法的本质区别：

1. GET用于信息获取，它是安全的（这里安全的含义是指非修改信息），而POST是用于修改服务器上资源的请求；
2. GET请求的数据会附在URL之后，而POST把提交的数据则放置在HTTP实体的主体里，所以，POST的安全性要比GET的安全性高；
3. GET方式提交数据和POST方法提交数据并没有限制数据大小。

## 其他方法

### HEAD

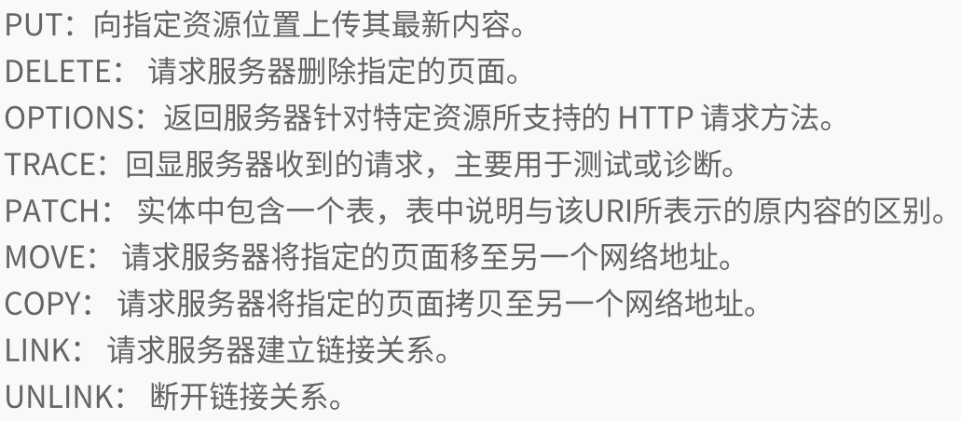
HEAD请求方法只请求页面的首部



HEAD和GET请求方法的区别：GET请求回来的报文有实体的主体部分，而HEAD请求回来的报文没有实体的主体部分。



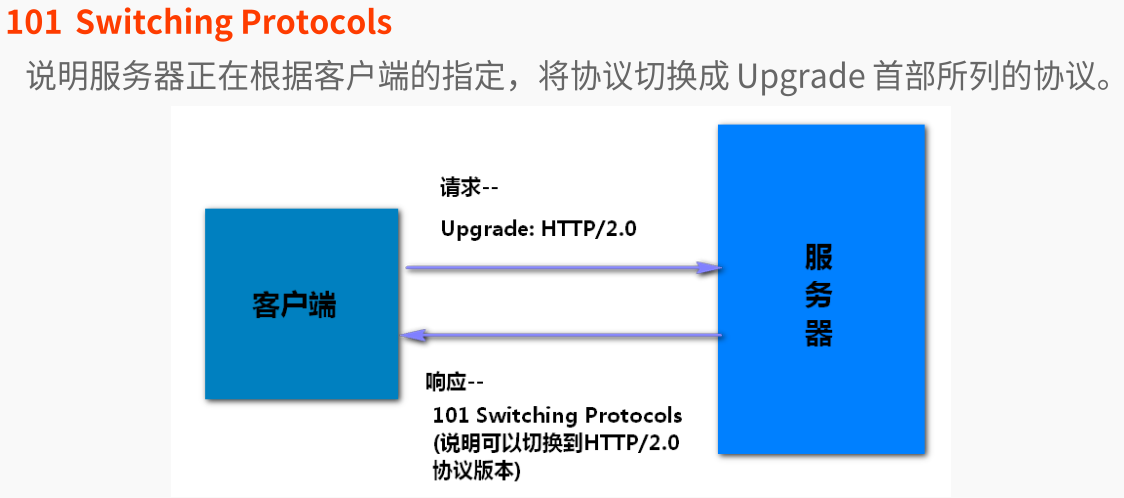
### 其他



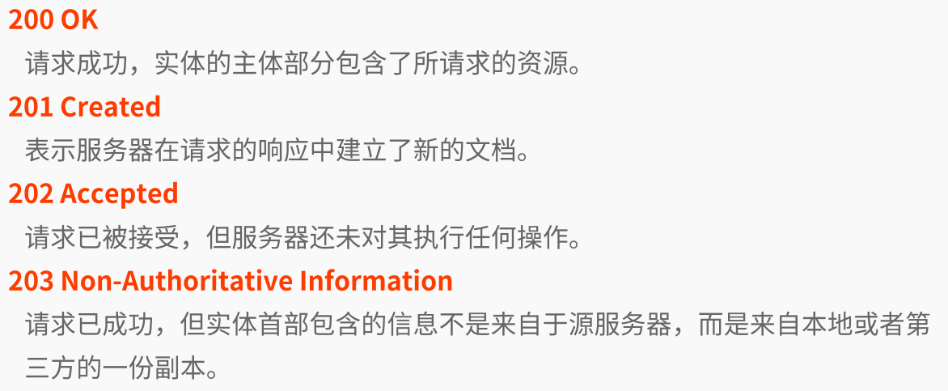
# HTTP状态码

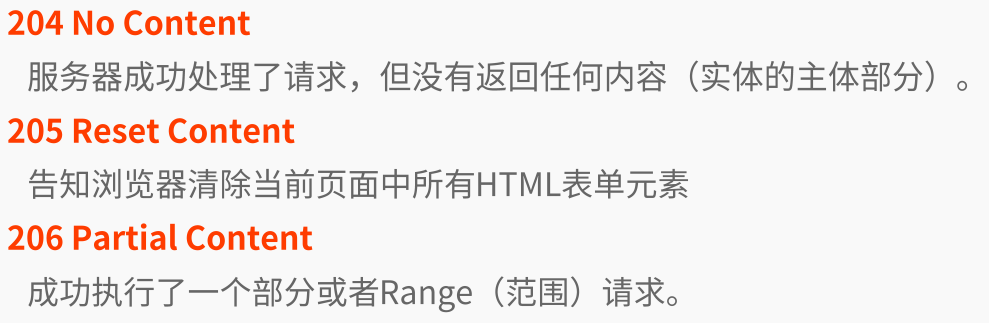
## 信息性状态码



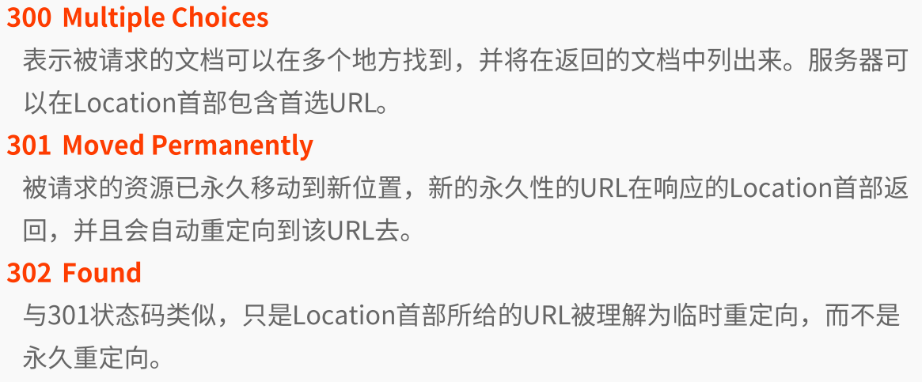


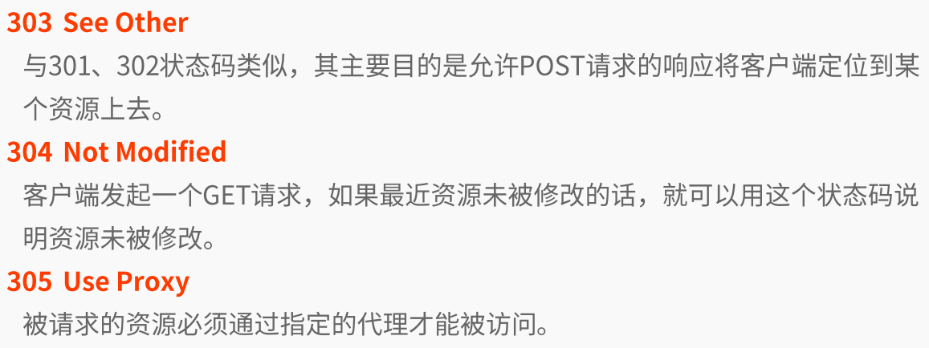
## 成功状态码





## 重定向错误状态码







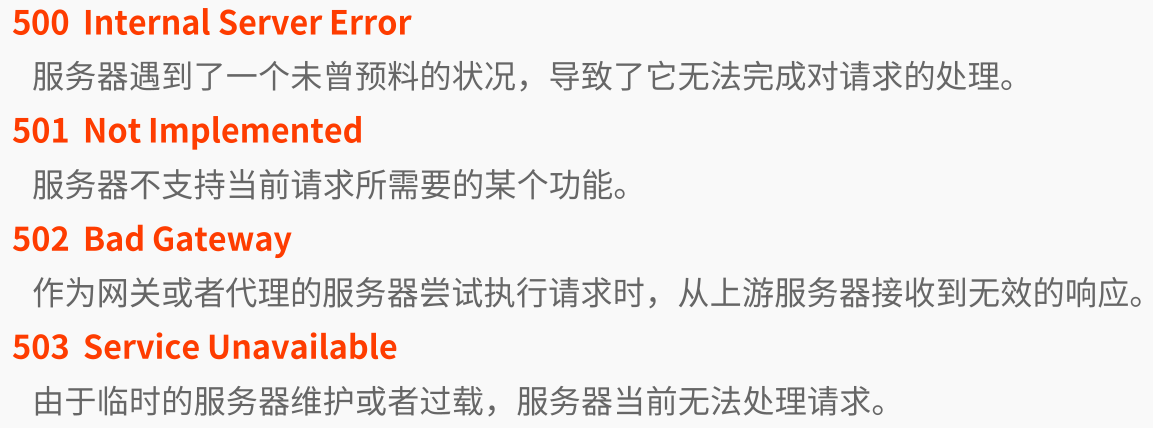
## 客户端错误状态码

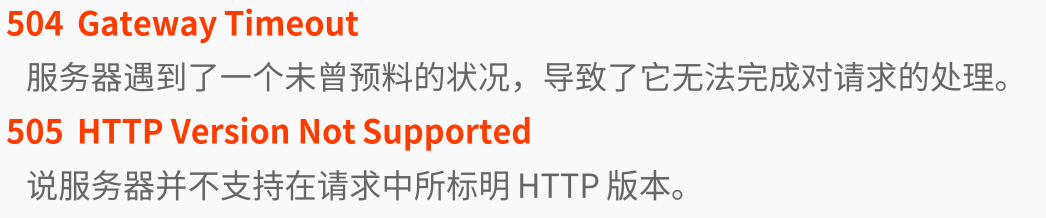






## 服务器错误状态码





# 报文格式

# 工作流程