

# 全国计算机等级考试二级

## 《WEB程序设计》

杨琦

**yangqi@mail.xjtu.edu.cn**

# 第二章 HTTP协议基础



HTTP的基本概念与交互模型

HTTP请求消息

HTTP响应消息

HTTP消息头

多用途Internet邮件扩展

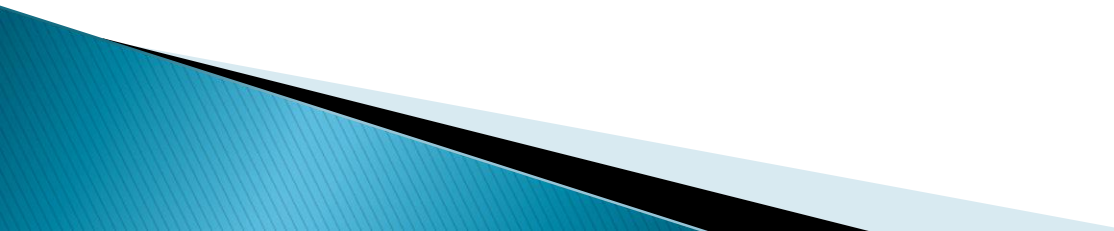
## 2.1 HTTP的基本概念与交互模型

- HTTP是一种通用的、无状态（Stateless）的应用层协议，基于标准的客户机/服务器模型。
- HTTP在可靠的网络协议（TCP/IP）的基础上提供了在Web服务器和客户机之间传输信息的一种机制，并制定了客户机与服务器之间交互的各种消息格式。

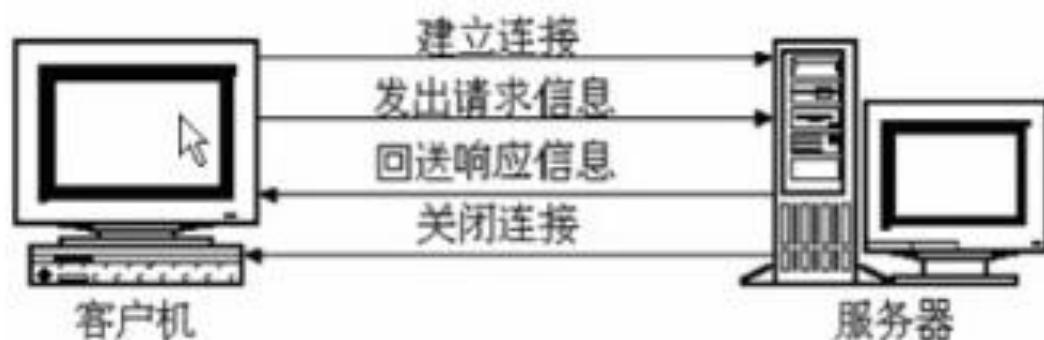
## 2.1.1 HTTP协议简介

- 0.9 已过时。只接受 GET 一种请求方法，没有在通讯中指定版本号，且不支持请求头。由于该版本不支持 POST 方法，所以客户端无法向服务器传递太多信息。
- HTTP/1.0 这是第一个在通讯中指定版本号的HTTP 协议版本，至今仍被广泛采用，特别是在代理服务器中。
- HTTP/1.1 当前版本。持久连接被默认采用，并能很好地配合代理服务器工作。还支持以管道方式同时发送多个请求，以便降低线路负载，提高传输速度。

# HTTP协议的特点：

- 1 支持客户机/服务器模型
  - 2 采用“请求/响应”的交互模式
  - 3 协议设计合理、简单、高效
  - 4 协议设计灵活，扩展性好
  - 5 无状态
  - 6 持久连接
  - 7 支持内容协商机制
- 

## 2.1.2 HTTP协议交互模型



四个步骤:

1. 客户端与服务器端建立连接
2. 客户端向服务器提出请求
3. 如果请求被接受，则服务器送回响应，在响应中包括状态码和所需文件
4. 断开客户端和服务器的连接

# HTTP交互模型中有两个关键点：

- 客户机在发送请求之前需要主动向服务器建立连接，服务器无法主动向客户机发起连接。
- 在客户机向服务器发送HTTP请求之后，服务器才能返回响应消息，服务器也无法主动向客户机发送数据。

# HTTP代理

- HTTP代理本质上是一个既作服务器又作客户机的中介程序，它的主要工作就是接收客户机发送的请求消息，转发给服务器，然后接收服务器返回的响应消息，再转发给客户机。





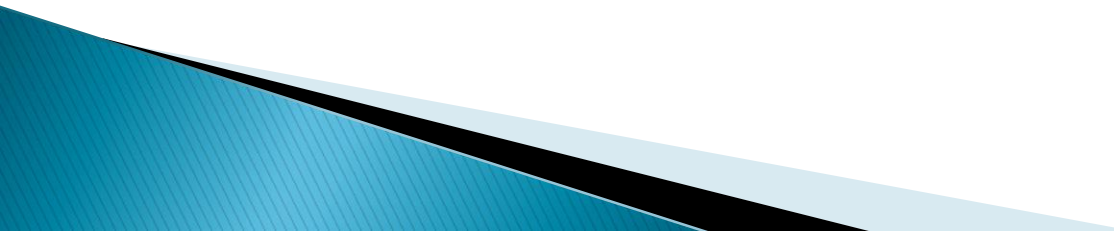
# HTTP代理的主要交互过程：

- 客户机建立与代理的TCP连接，并发送请求消息。
- 代理分析收到的请求消息，如果能自己处理，则直接生成响应消息发送给客户机，否则与服务器建立TCP连接，把经过翻译处理的HTTP请求消息发送给实际的服务器。
- 服务器分析处理请求消息，生成HTTP响应消息，返回给代理，并在对方确认收到后关闭连接。
- 代理接收到响应消息后，然后把响应再发给客户机，经过对方确认收到后关闭连接。

## 2.1.3 HTTP消息格式

- HTTP消息格式有两种类型：请求消息和响应消息
- 无论是请求消息，还是响应消息都有：
- 开始行（start-line）或状态行（status-line）
- 消息头（headers）
- 消息体（message-body）

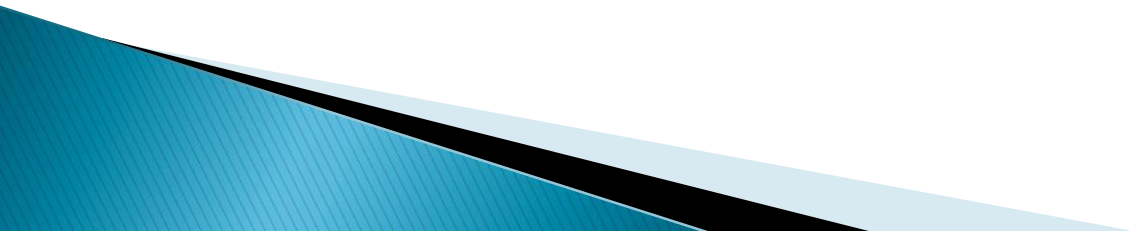
# 客户机的请求消息：

1. GET /hello.html HTTP/1.1
  2. Host: test.website.com
  3. Accept: \*/\*
  4. Connection: keep-alive
  5. Cache-Control: no-cache
  6. User-Agent: Mozilla/4.04[en] (Win95;I;Nav)
  - 7.
- 

# 服务器返回响应

1. HTTP/1.1 200 OK
2. Content-Type: text/html
3. Content-Length:55
4. Connection:keep-alive
- 5.
6. <html><body>
7. <h1>This is for test.</h1>
8. </body></html>

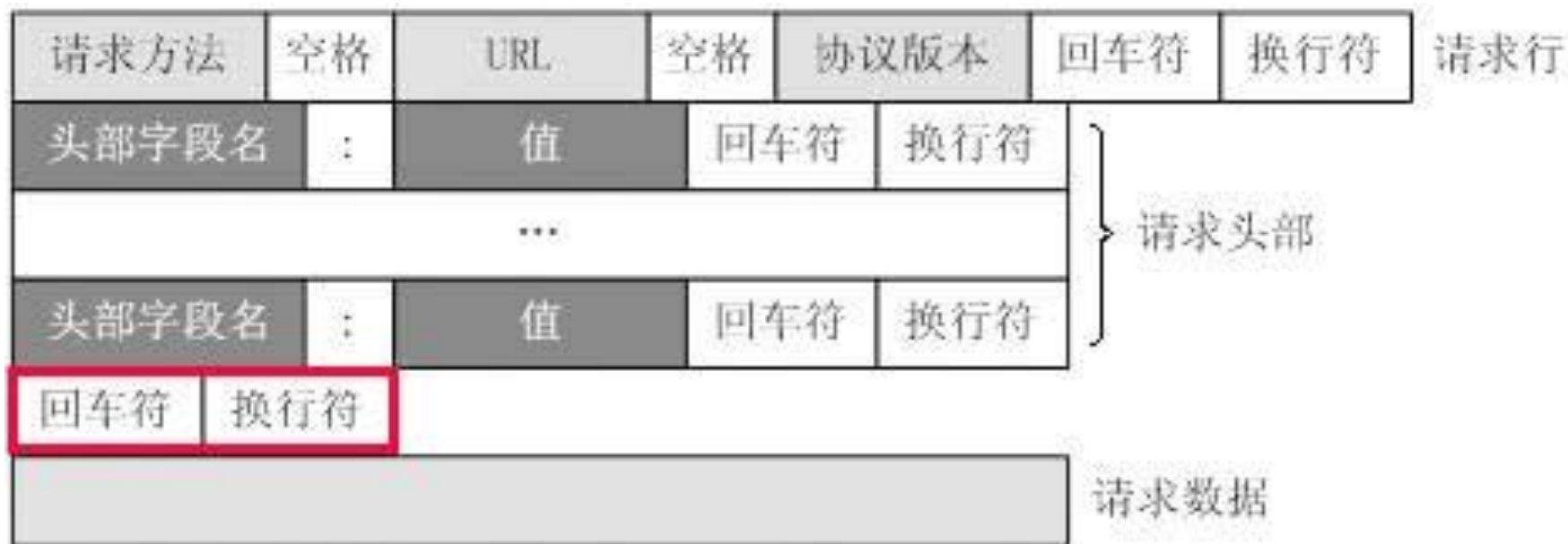
## 2.2 HTTP请求消息



## 2.2.1 HTTP请求消息格式

- 开始行
- 消息头
- 消息体

## 2.2.1 HTTP请求消息格式



# 典型HTTP请求消息例子

POST /action/login.do HTTP/1.1

Host: login.website.com

Content-Length:29

Content-Type:application/x-www-form-urlencoded

User-Agent: Mozilla/5.0 (Windows NT 6.1)Gecko/ Firefox/16.0

Accept:\*. \*

Accept-Language:zh-ch, en-us

Accept-Encoding:gzip, deflate

Connection:Keep-alive

Refer:http://login.website.com/form/loginform.html

user=testuser&password=123456



## 2.2.2 HTTP请求方法

方法	描述
GET	对服务器资源的简单请求
HEAD	类似于get请求，只不过返回的响应中没有具体的内容，用于获取报头
POST	用于发送包含用户提交数据的请求
PUT	传送当前请求文档的一个资源
DELETE	发送一个用来删除指定文档的请求
TRACE	发送请求的一个副本，以跟踪其处理进程
OPTIONS	返回所有可用的方法；可检查服务器支持哪些方法
CONNECT	用于ssl隧道的基于代理的请求

# HTTP请求消息例子

- 一个URL为“http://test.com/ask.asp?name=liyang”的GET请求报文例子如下：

GET http://test.com/ask.asp?name=liyang HTTP/1.1

Accept: \*/\*

Accept-Language: zh-cn

Accept-Encoding: gzip, deflate

User-Agent: Mozilla/4.0 (compatible; MSIE 6.0;)

Windows NT 5.1; SV1; .NET CLR 2.0.50727)

Host: www.test.com

Connection: Keep-Alive

## 2.2.3 HTTP请求URI

- URI : Uniform Resource Identifier, 统一资源标识符;
- URI常见有两种形式: **URL和URN**
- URL: Uniform Resource Locator, 统一资源定位符;
  - URL分为两大类, **绝对URL和相对URL**。
- URN: Uniform Resource Name, 统一资源名称。

## 2.3 HTTP响应消息

- 开始行
- 消息头
- 消息体

## 2.3.1 HTTP响应消息格式

下面是一个典型的HTTP响应的例子：

1. HTTP/1.1 200 OK
2. Content-Length:55
3. Content-Type:text/html
4. Keep-Modified:Sat, 01 Jan 2011 11:0518 GMT
- 5.
6. <html><body>
7. <h1>This is for test.</h>
8. </body></html>

## 2.3.2 HTTP--响应状态码

1xx: 普通信息码——表示请求已接收，继续处理

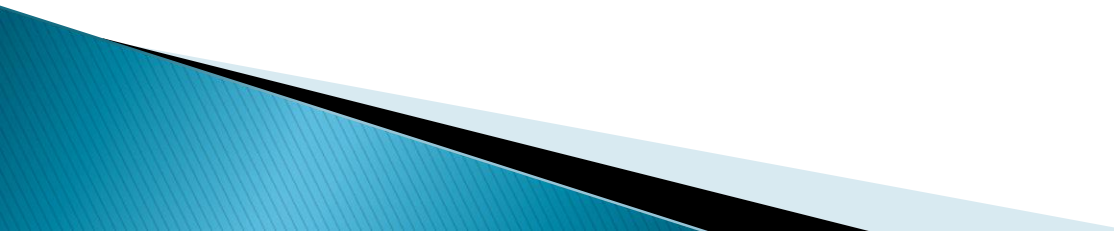
2xx: 操作成功码——表示请求已经被成功接收，理解，接受

3xx: 重定向码——要完成请求必须进行更进一步的操作

4xx: 客户机错误码——请求有语法错误或请求无法实现

5xx: 服务器错误码——服务器未能实现合法的请求

## 2.4 HTTP消息头

- 内容协商消息头
  - 缓存控制消息头
  - 实体描述消息头
  - 条件控制消息头
  - 其他消息头
- 

## 2.4.1 内容协商消息头

常见的内容协商消息头字段主要有

- Accept,
- Accept-Charset,
- Accept-Encoding,
- Accept-Language
- Allow。



## 2.4.2缓存控制消息头

- (1) Cache-Control字段：常见的可能取值有，no-cache, no-store以及max-age。
- (2) Pragma字段。取值为no-cache。

## 2.4.3 实体描述消息头

(1) Content-Type 字段。用于描述消息中数据实体的媒体类型。

(2) Content-Length 字段。用于描述消息中数据实体的大小，以字节数表示。

## 2.4.4 条件控制消息头

- (1) If-Match字段。用于请求消息客户机使用本字段控制服务器执行条件判断所依据的对象是请求操作针对的数据实体。
- (2) If-Modified-Since字段。用于请求消息客户机使用本字段控制服务器执行条件判断所依据的对象是请求操作针对的数据实体。
- 例如：If-Modified-Since:Sat, 29 Oct 1994 19:43:31 GMT

## 2.4.5其他消息头

(1)Host字段。用于请求消息知名客户机请求的资源所在的主机及其端口号。端口号如果省略，则表示使用默认端口80。Host:www.test.com

(2)Location字段。用于响应消息。主要用于在3xx状态码的响应消息中指明重定义后的URL在201created状态码的响应消息中，Location指明了新创建资源的URI。

Location:http://www.w3.org/pub/index.html

(3)Date字段。用于请求消息和响应消息表示消息产生的时间。

(4>Last-Modified字段。用于请求消息和响应消息描述消息体中数据实体最后被修改的时间。

(5>User-Agent字段。用于请求消息，它的取值描述了发出请求的客户机的相关信息，该信息可以包括客户机的名称，版本号，所运行的操作系统平台等。

User-Agent: Mozilla/4.0 (compatible; MSIE 6.0;)

(6)Referer字段。用于请求自带它用于向服务器指明本次请求的URL是从哪个URI资源中获得的。

Referer:http://www.w3.org/hypertext/DataSources/Overview.html

## 2.5 多用途Internet邮件扩展

- MIME (Multipurpose Internet Mail Extensions)  
多用途互联网邮件扩展类型。
- 最早应用于电子邮件系统
- 后引入到HTTP协议中，用于描述所传输的多媒体类型，使得HTTP不仅可以传输普通的文本数据，还可以传输丰富的多媒体数据和其他二进制数据。

## 2.5.1 MIME简介

- 电子邮件协议与HTTP协议格式非常相似。
- 电子邮件由信头和信体（body）构成。
- 信头由一系列的头字段组成。
- 头字段的格式为：
  - <字段名><: ><字段的值><CRLF>

## 2.5.1 MIME简介

- MIME增加的头字段:
  - MIME-Version:
  - Content-Description:
  - Content-ID:
  - Content-Transfer-Encoding:
  - Content-Type:
  - .....
- 

# Content-Type

- Content-Type的值得格式为：类型/子类型
- 独立媒体类型（Discrete Media Type）
  - text/plain
  - text/richtext
  - image/gif
  - image/jpeg
- 复合媒体类型（Composite Media Type）
  - message/rfc822
  - message/partial
  - message/external-bldy



## 2.5.2 MIME在HTTP协议中的应用

- MIME-Version:1.0
- Content-Type: multipart/related;
- type="multipart/alternative";
- boundary="====\_ABC1234567890DEF\_===="
- --====\_ABC1234567890DEF\_====
- Content-Type: multipart/alternative;
- boundary="====\_ABC0987654321DEF\_===="
- --====\_ABC0987654321DEF\_====
- Content-Type: text/html;
- charset="iso-8859-1"
- Content-Transfer-Encoding: 7bit
- <HTML><HEAD></HEAD><BODY bgColor=#ffffff>
- <iframe src=cid:EA4DMGBP9p height=0width=0>
- </iframe></BODY></HTML>
- --====\_ABC0987654321DEF\_====--
- --====\_ABC1234567890DEF\_====

# 结 束 语

- 学好程序设计语言的唯一途径是

上机练习

- 你的编程能力与你在计算机上投入的时间成

正比