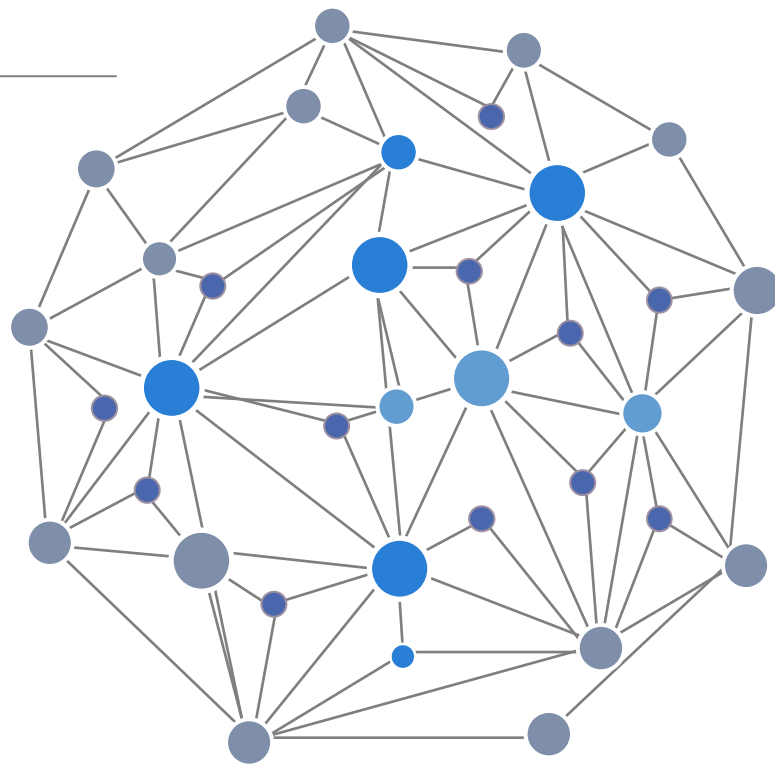

Web 程序设计

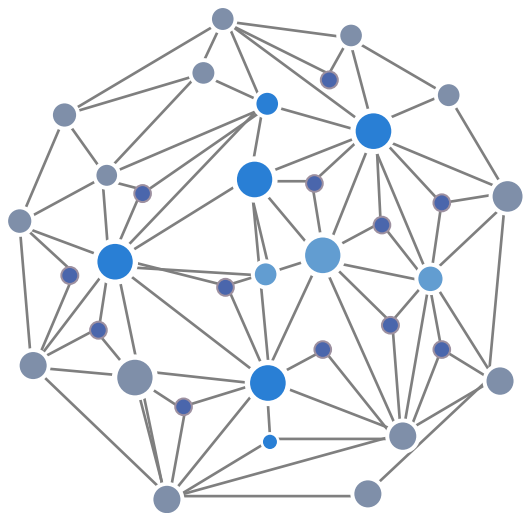
第九讲 HTTP 协议与 Web 服务器

福州大学 计算机与大数据学院
软件工程系 陈昱



内容提要

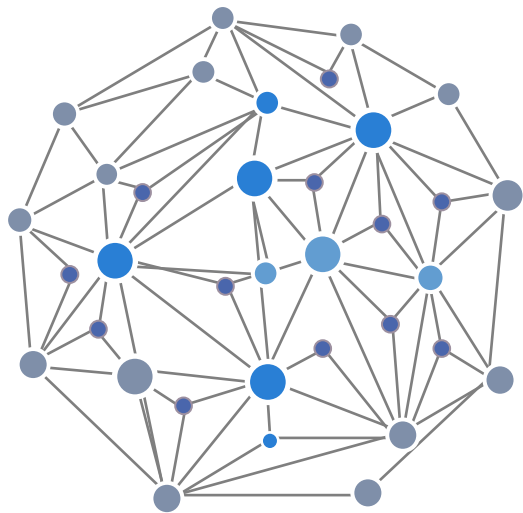
- 服务器端相关技术
- HTTP 协议及其相关标准
- Web 浏览器
- Web 服务器
- CGI 技术与 Perl
- 参考资料



HTTP 协议及其相关标准

与 HTTP 协议相关的技术

- 统一资源定位符 URL
- 多用途因特网邮件扩充协议 MIME



统一资源定位符 URL

Uniform Resource Locators

RFC 1738

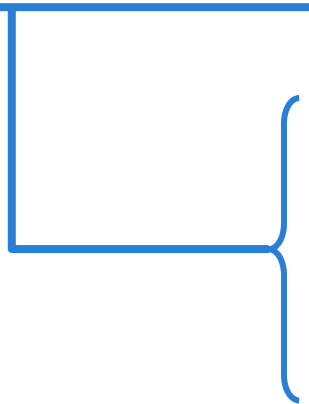
统一资源定位符 URL

- 怎样标识分布在整个因特网上无数的万维网文件?
- 使用 **统一资源定位符 URL** (Uniform Resource Locator) 来标识万维网上的文档
- 使每一个资源在整个因特网的范围内具有唯一的标识符: **URL**

URL 的一般形式

- 由 (://) 隔开的两大部分组成
- URL 的一般形式是：

<URL的访问协议>://<主机名>:<端口>/<路径>/<文件名>



http	——	超文本传送协议 HTTP
ftp	——	文件传送协议 FTP
mms	——	微软媒体服务器协议

HTTP 使用的 URL：主机名

- HTTP使用的 URL 的一般形式

http://<主机>:<端口>/<路径>/<文件名>

↑
这里是服务器的主机名 Hostname

- **域名** *DomainName* 是在域名商处注册，并可以通过全球 **DNS** 系统解析的全球唯一的主机名

DNS (Domain Name System)

- 域名系统是将域名和 IP 地址相互映射的一个分布式数据库
- 能够使人更方便的访问互联网，而不用去记忆被机器使用的 IP 数字
- 在互联网上我们可以通过付费在域名商那里注册域名，通过其 DNS 管理系统将域名与我们的服务器 IP 绑定
- DNS 服务使用 TCP 和 UDP 端口 53

HTTP 使用的 URL：端口号

- HTTP使用的 URL 的一般形式

http://<主机>:<端口>/<路径>/<文件名>

HTTP 的默认端口号是 80，通常可省略；
如果不是 80，则必须标注出来，例如 8080

HTTP 使用的 URL：路径信息

- HTTP使用的 URL 的一般形式

http://<主机>:<端口>/<路径>/<文件名>

若省略文件的<路径>/<文件名>项，则
URL 就指向网站服务器的某个默认文件

- 默认文件名通常为 index.html, index.htm, index.php 等

HTTP 使用的 URL：查询字符串

`http://zh.wikipedia.org:80/wiki/Search?search=铁路&go=Go`

? 后面的部分是查询(Query)，是客户端发送给服务器端程序的参数，参数之间用 & 分隔

使用 HTTP 的 URL：段标识符

<http://www.oreilly.com/cgi/calender.cgi?month=july#week3>



后为段标识符，指向资源中的一个特定段

- 段标识符不会发送给服务器
- 对于 XHTML 文件来说，段标识符相当于页内**锚点** (anchor)，浏览器会跳转到文件中设置了相应的 id 属性值的元素处（内部跳转）

URL Character Encoding

- 如果 URL 出现中文该怎么办？如何传输中文的参数给服务器端的程序？
- 解决方法：对 URL 进行编码
- 将字符转换成十六进制编码，前面加上 %
- “汉” 字，UTF-8 编码为 0xE6 B1 89
则在 URL 中写做：%E6%B1%89
- “汉” 字，GBK 编码为 0x BABA
则在 URL 中写作：%BA%BA

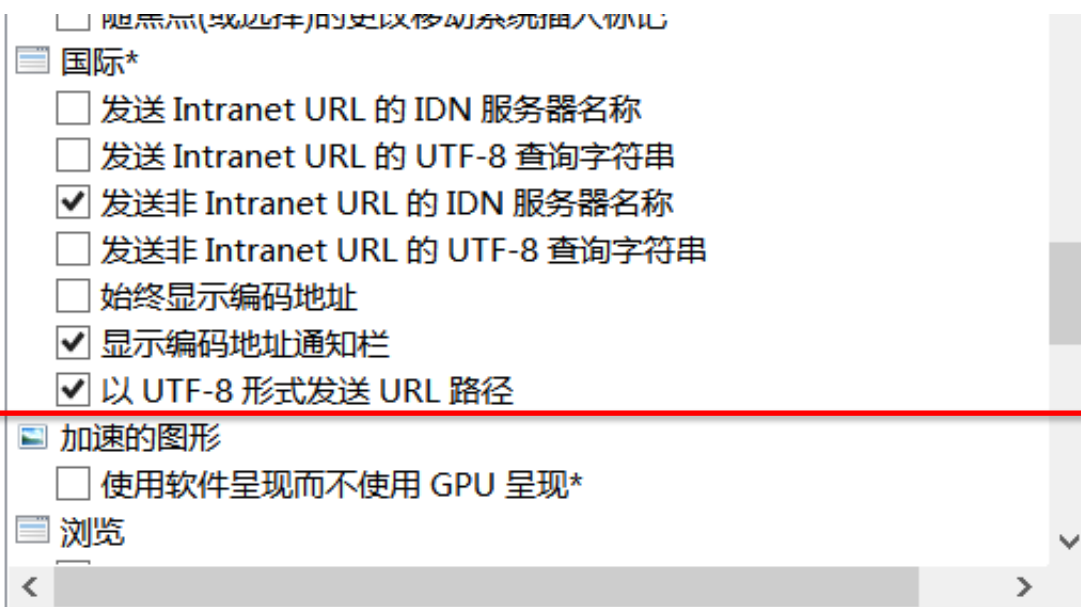
如果直接在地址栏输入中文呢？

- 到底传输给服务器端的是UTF-8还是GBK？

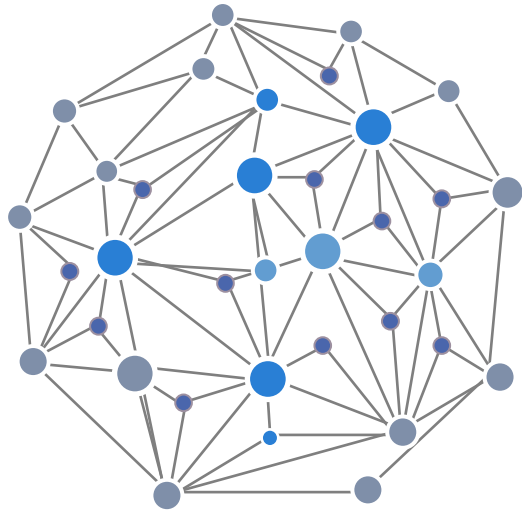


URL Character Encoding

- 如果在浏览器的地址栏中对中文没有使用 URL 编码方式进行明确编码，则由浏览器自行决定用何种编码发送
 - 可能是 GBK，也可能是 UTF-8



*在重新启动你的计算机后生效



MIME Type

Multipurpose Internet Mail Extensions: Media
Type (RFC 2046)

多用途因特网邮件扩充协议 MIME

- 通过互联网传输的文件类型如此之多（文本，图像，音频，视频...），每种文件通常需用不同的方式进行处理显示
- 把 JPEG 当成 PNG 图像来解码肯定是不对
- 浏览器收到数据后，如何知道数据的正确类型（格式）并正确显示呢？

多用途因特网邮件扩充协议 MIME

- **方法**：服务器在传输文件给浏览器的时候，同时告诉浏览器该文件的文件类型
- 这就是 **MIME Type** 的作用
- 现在它已经演化成一种指定文件类型的通用方法
 - 最早用于 SMTP，后来 HTTP 和 SIP 也用
 - 新的名称叫 Internet media type

MIME Media Type

- MIME Type 声明格式如下：
类型(type)/子类型(subtype)
- 例如：
 - HTML 文档的类型标识为：**text/html**
 - CSS 代码的类型标识为：**text/css**
 - JS 程序的类型标识为：**text/javascript**
 - JPEG 图像的类型标识为：**image/jpeg**

MIME Media Type

- Text
 - 子类型: plain, html 等
- Image
 - 子类型: jpeg, gif 等
- Audio
 - 子类型: basic (8-bit mu-law encoded), mid , wav 等
- Video
 - 子类型: mpeg, quicktime 等
- Application
 - 需使用其他阅读器的数据
 - 子类型: msword, pdf 等

MIME Media Type 与浏览器

- Firefox中用 about:plugins 作为地址，可以查看各种插件支持的 MIME 类型



已安装的插件

少些东西？某些插件已不再支持。 [详细了解...](#)

OpenH264 视频编码器，由思科系统公司提供

文件： 1.7.1
路径： C:\Users\cheny\AppData\Roaming\Mozilla\Firefox\Profiles\gsdmjhob.default\gmp-gmpopenh264\1.7.1
版本： 1.7.1
状态： 启用
此插件由 Mozilla 自动安装，以遵从 WebRTC 技术标准，满足设备进行 WebRTC 通话所需要的 H.264 视频编码器。访问 <http://www.openh264.org/> 可查看编码器的源代码和详细了解其实现。

MIME 类型	描述	后缀
---------	----	----

Widevine 内容解密模块，由 Google 公司提供

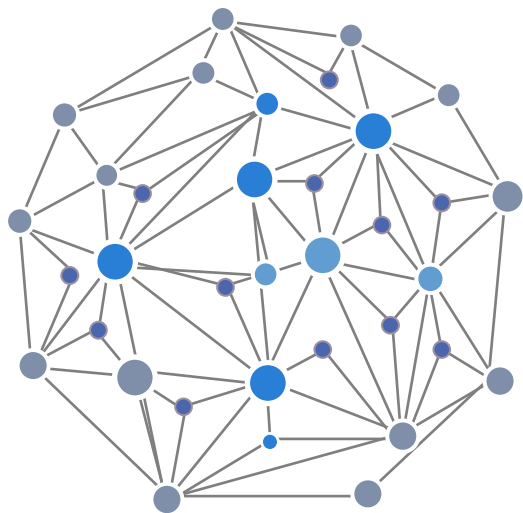
文件： 1.4.8.903
路径： C:\Users\cheny\AppData\Roaming\Mozilla\Firefox\Profiles\gsdmjhob.default\gmp-widevinecdm\1.4.8.903
版本： 1.4.8.903
状态： 启用
播放受保护的网络视频。

MIME 类型	描述	后缀
---------	----	----

Shockwave Flash

文件： NPSWF64_26_0_0_131.dll
路径： C:\Windows\system32\Macromed\Flash\NPSWF64_26_0_0_131.dll
版本： 26.0.0.131
状态： 启用 (STATE_VULNERABLE_UPDATE_AVAILABLE)
Shockwave Flash 26.0 r0

MIME 类型	描述	后缀
application/x-shockwave-flash	Adobe Flash movie	swf
application/futuresplash	FutureSplash movie	spl



HTTP 协议

WWW 的基础协议

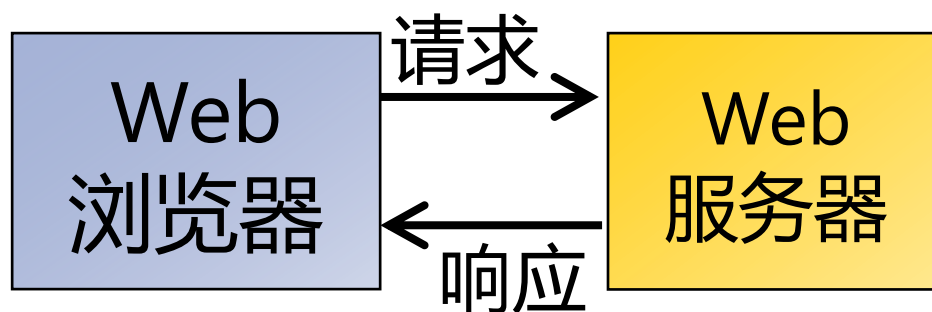
RFC 2616

深入理解 HTTP 协议的历史和演变过程，对于设计高性能的网站体系架构能够起到显著的作用

—— 摘自《程序员》2008 第5期

HTTP: HyperText Transfer Protocol

- 超文本传输协议
 - WWW (Web) 的基础
- 客户端/服务器模式
 - 客户端: 浏览器请求、接收、展示 Web 内容
 - 服务器: Web 服务器对请求进行响应,发送内容



HTTP 协议的工作流程

HTTP 是应用层协议，工作在 TCP 之上：

1. 客户端发起 TCP 连接到服务器, 端口 80
2. 服务器接受来自客户端的 TCP 连接
3. HTTP 报文 (应用层协议报文) 在浏览器 (http client) 和Web服务器 (http server) 之间进行交换
4. 关闭 TCP 连接

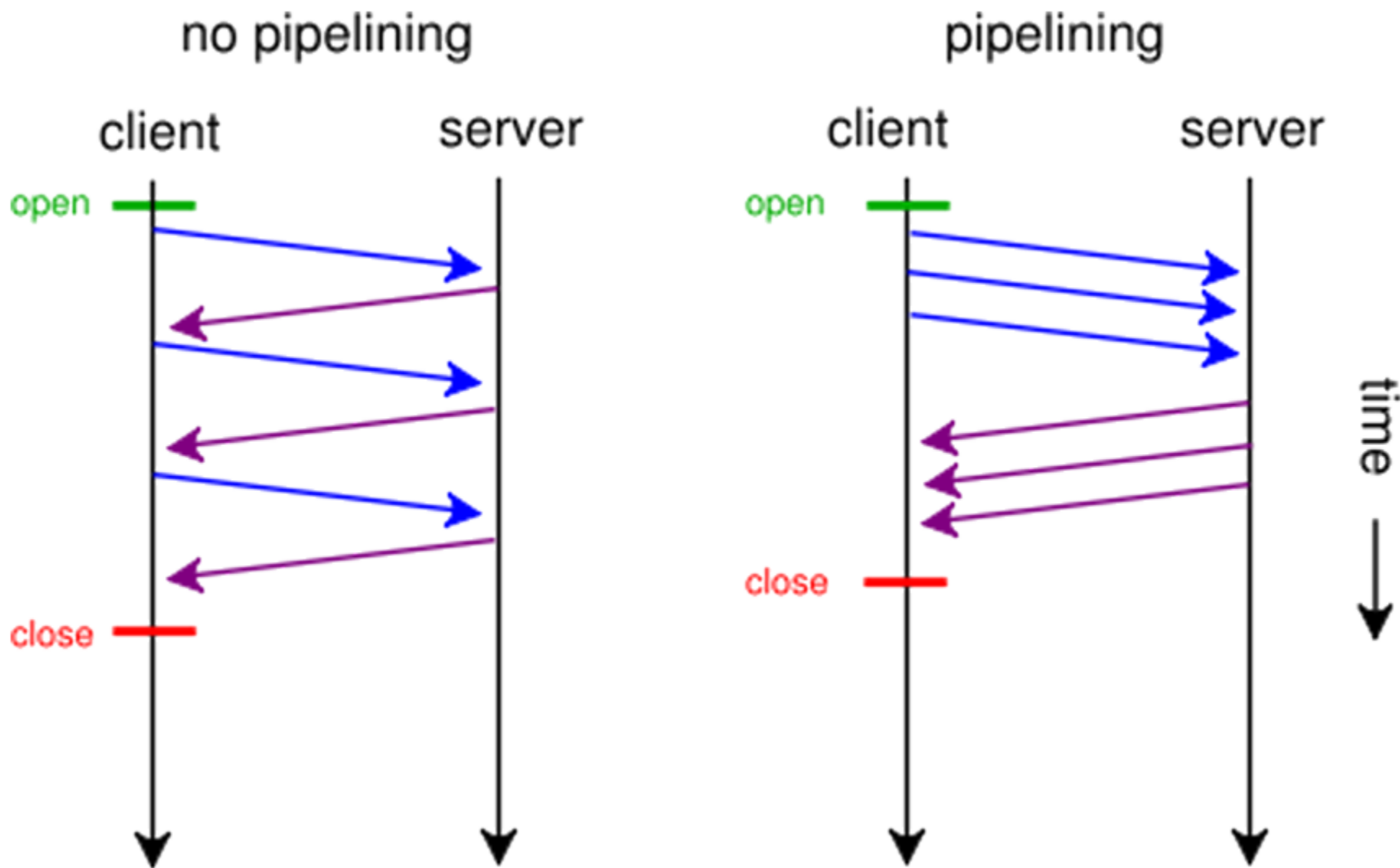
HTTP 协议的版本

- HTTP/0.9
 - 古老的版本
 - 1991, Tim Berners-Lee
- HTTP/1.0
 - 1995, 开始成为重要的面向事务的应用层协议
- HTTP/1.1
 - 1999, 目前最常用的版本
 - 增加了持久连接, Cache 等新功能

持久连接

- HTTP/1.0 非持久连接
 - 每个请求/响应需要一个新的 TCP 连接
 - 下载一个带有多个图片的网页，需要多次连接
 - 对用户请求响应慢、网络拥塞严重
- HTTP/1.1 支持持久连接（流水线）
 - 客户端可以在一个 TCP 连接中发起多个请求，而不必等待多个响应；例如通过一次连接请求页面上多个图片

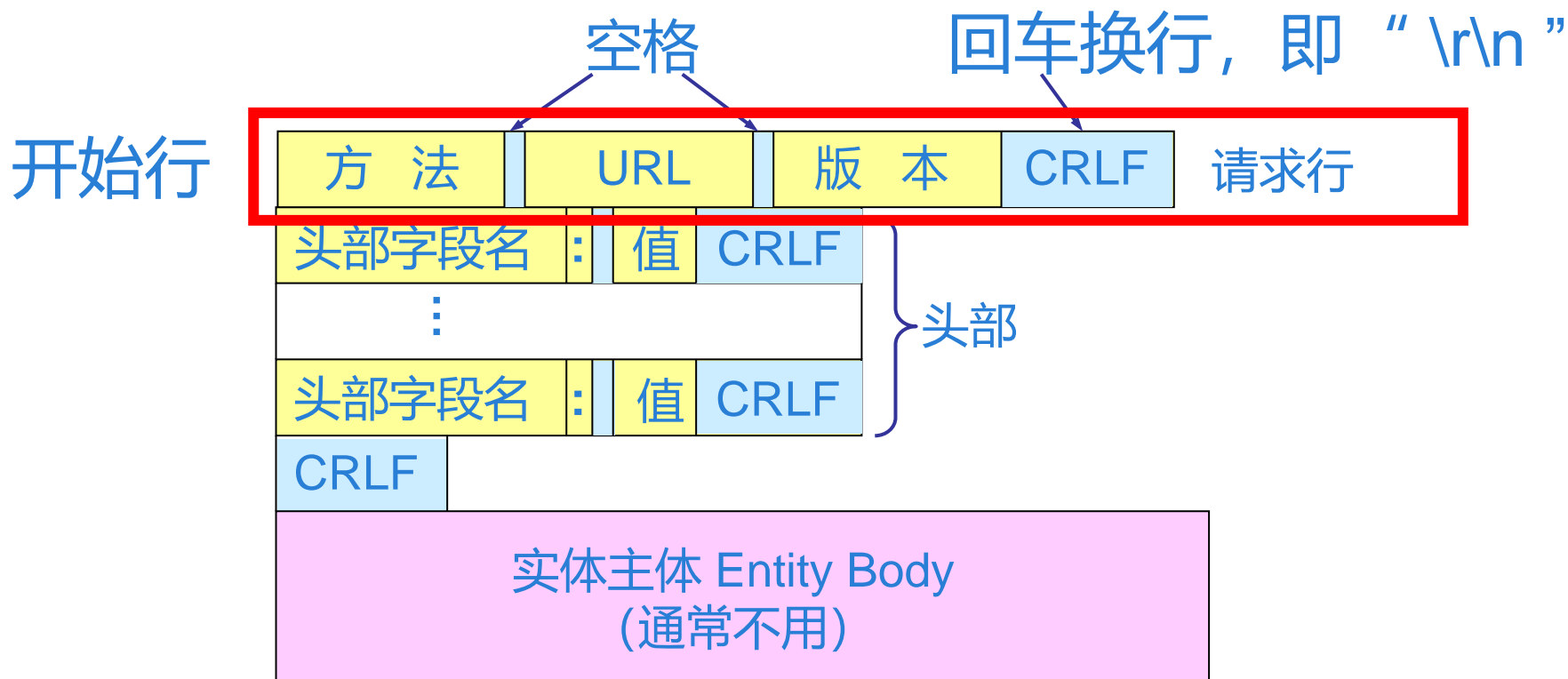
非流水线 vs 流水线方式



HTTP 的协议报文结构

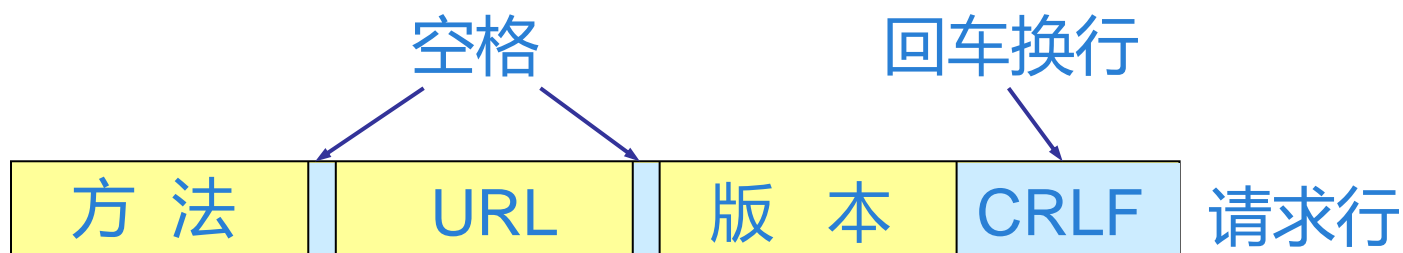
- HTTP 有两类报文: ***Request, Response***
- **请求报文:** 客户端向服务器发送的请求信息
- **响应报文:** 服务器给客户端的回应信息

HTTP 请求报文结构



报文由三个部分组成, 即开始行、头部行和实体主体
在请求报文中, 开始行称为请求行

HTTP 请求报文 — 请求行



- “方法” 就是对所请求的对象进行的操作
- “URL” 是所请求的资源的 URL
- “版本” 是 HTTP 的版本

HTTP 请求方法

方法	意义
GET	向服务器请求指定的资源
HEAD	只返回HTTP头部而没有内容
POST	请求服务器修改存储在服务器上的信息
PUT	请求服务器在服务器上生成和替换资源
DELETE	请求服务器删除服务器上的资源
OPTION	请求服务器列出对指定资源所有可用的请求方法
TRACE	请求服务器在收到请求头部后返回它
CONNECT	用于SSL和代理服务器

GET 方法

- 最简单的GET方法报文:

GET URL HTTP/1.1

Host: HostName

- 请求行后跟随一些可选的**字段**
 - 字段由**字段名**和**字段值**构成
 - 字段名和字段值之间用冒号隔开
 - Host 字段对于 HTTP 1.1 是必须的

Firefox 发起连接到 google.com

- GET / HTTP/1.1

Host	www.google.com.hk
User-Agent	Mozilla/5.0 (Windows NT 6.1; rv:15.0) Gecko/20100101 Firefox/15.0 FirePHP/0.7.1
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding	gzip, deflate
Accept-Language	zh-CN,zh;q=0.9,en-US;q=0.7,en;q=0.6,en-gb;q=0.4,en-us;q=0.3,en;q=0.1
Cache-Control	max-age=0
Connection	keep-alive

请求报文的头部信息

- Host 服务器主机名(服务器通过它支持虚拟主机)
- User-Agent 指明浏览器的版本
- Referer 告诉服务器用户从哪里跳转来
- Accept 浏览器支持的媒体类型
- Authorization
可以发送用户名、密码，用于访问受限的文档

.....

User-Agent

- User-Agent 通常表明了浏览器的种类
- 服务器可以根据该信息统计各种浏览器的使用情况
 - IE 6: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
 - Firefox 3: Mozilla/5.0 (Windows; U; Windows NT 5.1; zh-CN; rv:1.9.0.10) Gecko/2009042316 Firefox/3.0.10
 - Opera 9.6: Opera/9.60 (Windows NT 5.1; U; zh-cn) Presto/2.1.1

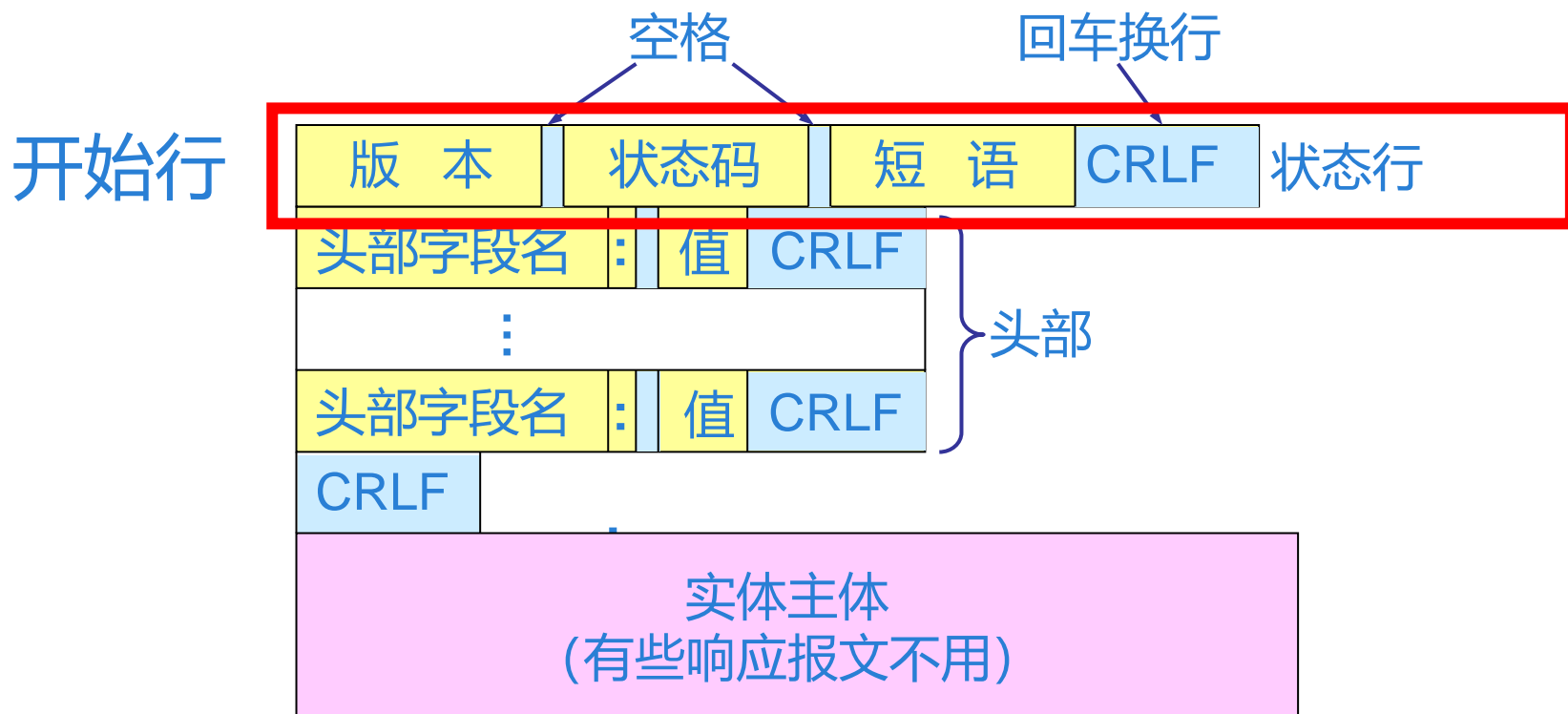
User-Agent

- 通过以下网站可以看到自己浏览器的 User-Agent
 - <http://www.useragent.org/>
- 可以用 Firefox 的扩展 “伪造” :
 - User Agent Switcher

Referer

- 当我们在网页上点击一个超链接跳转到另一个页面时，这时候的 HTTP 报文中会多了一个字段 Referer
- 比如在 google 首页点击 "google大全"
- Referer: <http://www.google.cn/>
- Referer 告诉服务器用户从哪里来

HTTP 的响应报文结构



响应报文的开始行是**状态行**。
状态行包括三项内容，即 **HTTP 的版本**，**状态码**，以及解释状态码的**简单短语**。

状态码都是三位数字

- 1xx 表示通知信息，如请求收到了或正在进行处理。
- 2xx 表示成功，如接受或知道了。
- 3xx 表示重定向，表示要完成请求还必须采取进一步的行动。
- 4xx 表示客户的差错，如请求中有错误的语法或不能完成。
- 5xx 表示服务器的差错，如服务器失效无法完成请求。

常见状态码

- **200 OK** 请求已经成功处理
- 301 Moved permanently 文档已经转移到新地址
- **304 Not modified** Cache中的文档是最新的
- 400 Bad request 客户端请求存在错误
- 403 Forbidden 客户端不允许访问该资源
- **404 Not found** 文件未找到
- 500 Internal server error 服务器内部错误
- 503 Service unavailable 服务器过载(超负荷)

响应报文的其他头部信息

- Date 当前的日期/时间
- Server Web服务器信息
- Last Modified 请求文档的最近修改时间
- Expires 请求文档的过期时间
- Content-Type 连接的媒体类型
- Connect-length 数据的长度
- Connect-encoding 说明有无使用压缩技术
- Transfer-encoding 说明采用的编码变换类型
-

Google 返回给 Firefox 的信息

- HTTP/1.1 200 OK

Cache-Control	private, max-age=0
Content-Encoding	gzip
Content-Type	text/html; charset=UTF-8
Date	Sun, 09 Sep 2012 13:13:48 GMT
Expires	-1
Server	gws
Transfer-Encoding	chunked

```
josh@blackbox:~$ telnet en.wikipedia.org 80
Trying 208.80.152.2...
Connected to rr.pmtpa.wikimedia.org.
Escape character is '^]'.
GET /wiki/Main_Page http/1.1
Host: en.wikipedia.org
```

Request

```
HTTP/1.0 200 OK
```

Response headers

```
Date: Thu, 03 Jul 2008 11:12:06 GMT
```

```
Server: Apache
```

```
X-Powered-By: PHP/5.2.5
```

```
Cache-Control: private, s-maxage=0, max-age=0, must-revalidate
```

```
Content-Language: en
```

```
Vary: Accept-Encoding, Cookie
```

```
X-Vary-Options: Accept-Encoding;list-contains=gzip,Cookie;string-contains=enwikiToken;string-contains=enwikiLoggedOut;string-contains=enwiki_session;
```

```
string-contains=centralauth_Token;string-contains=centralauth_Session;string-contains=centralauth_LoggedOut
```

```
Last-Modified: Thu, 03 Jul 2008 10:44:34 GMT
```

```
Content-Length: 54218
```

```
Content-Type: text/html; charset=utf-8
```

```
X-Cache: HIT from sq39.wikimedia.org
```

```
X-Cache-Lookup: HIT from sq39.wikimedia.org:3128
```

```
Age: 3
```

```
X-Cache: HIT from sq38.wikimedia.org
```

```
X-Cache-Lookup: HIT from sq38.wikimedia.org:80
```

```
Via: 1.0 sq39.wikimedia.org:3128 (squid/2.6.STABLE18), 1.0 sq38.wikimedia.org:80 (squid/2.6.STABLE18)
```

```
Connection: close
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Response body

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en" dir="ltr">
```

```
<head>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

```
<meta name="keywords" content="Main Page,1778,1844,1863,1938,1980 Summer Olympics,2008,2008 Guizhou riot,2008 Jerusal
```

```
...
```

```
... This content has been removed to save space
```

```
...
```

```
"Non-profit organization">nonprofit</a> <a href="http://en.wikipedia.org/wiki/Charitable_organization" title="Charitable organization">charity</a>.<b
r /></li>
```

```
<li id="privacy"><a href="http://wikimediafoundation.org/wiki/Privacy_policy" title="wikimedia:Privacy policy">Privac
y policy</a></li>
```

```
<li id="about"><a href="/wiki/Wikipedia:About" title="Wikipedia:About">About Wikipedia</a></li>
```

```
<li id="disclaimer"><a href="/wiki/Wikipedia:General_disclaimer" title="Wikipedia:General disclaimer">Disclaimers</a>
```

```
</li>
```

```
</ul>
```

```
</div>
```

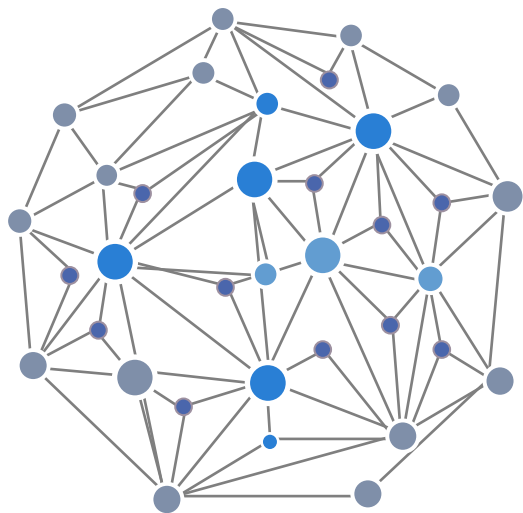
```
</div>
```

```
<script type="text/javascript">if (window.runOnLoadHook) runOnLoadHook();</script>
```

```
<!-- Served by srv93 in 0.050 secs. --></body></html>
```

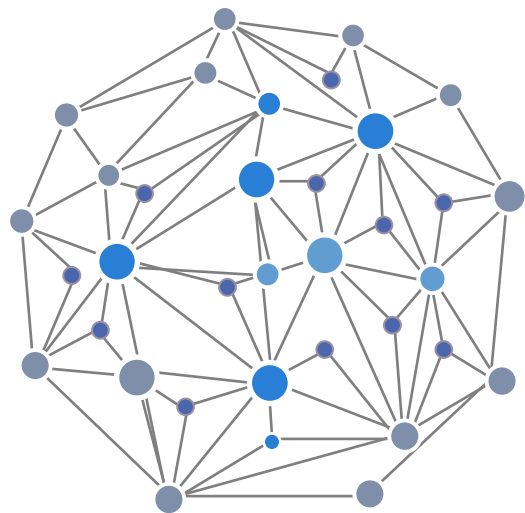
```
Connection closed by foreign host.
```

```
josh@blackbox:~$ █
```



HTTP 的一些功能介绍

cookie, cache, etag, gzip etc.



1、Cookie 机制介绍

有状态的浏览器/服务器交互

- 像 taobao.com 这样的网站似乎“知道我是谁”，它们是如何做到的？
- 用户如何让服务器识别自己的身份？
- 服务器又如何向每一个用户提供特定的内容？



Cookie

- HTTP是无状态协议：它只是简单地允许浏览器从服务器上请求一个资源
- Netscape 的解决方案: **cookie**
 - 少量的由服务器发送给浏览器的信息，并且会在未来的页面请求时发送回服务器。
- cookie 有很多用处: 身份验证，用户跟踪，保存用户选择、购物车等
- 一个cookie 的数据包含一个键/值对，在用户的 HTTP GET 或者POST 请求的header中传递

Cookie 工作流程图

1、浏览器请求一个 Web 页面



2、服务器发送页面 + cookie



Web
浏览器



Web
服务器

3、浏览器请求另一个页面



Cookie 先前浏览的一些信息

Cookie 实现原理

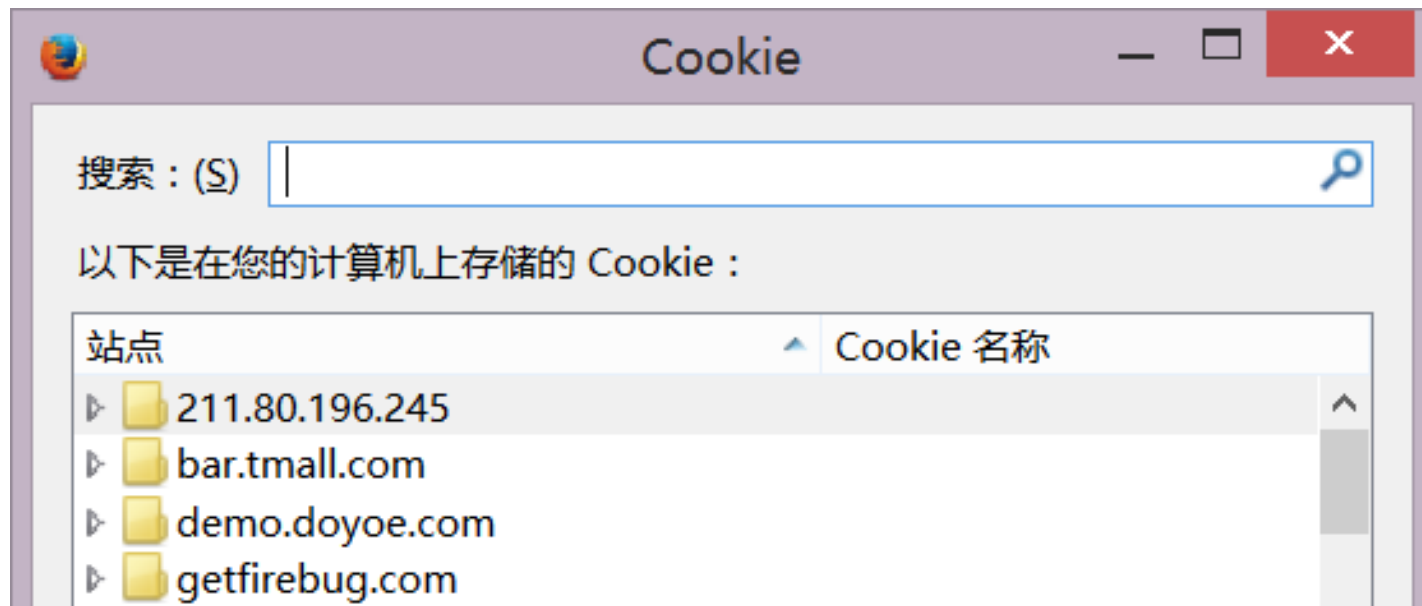
- 服务器可以在响应报文中使用 **Set-cookie** 在用户的机器上设置一个 cookie
 - Set-cookie: CUSTOMER=Dave_Reed; PATH=/; EXPIRES=Thursday, 29-Jan-04 12:00:00
- 当用户重新访问 PATH 指定的 URL 时, 浏览器通过请求报文发送 cookie 给服务器
 - Cookie: CUSTOMER=Dave_Reed

一个 cookie 能够存活多久?

- 会话 cookie：默认的类型；一个临时保存在浏览器内存的 cookie
 - 当浏览器关闭的时候, 临时的 cookie 将会被删除
 - 不能用于追踪长期信息
 - 比较安全, 因为除浏览器外没有程序能够访问它们
- 长期 cookie：能够在浏览器的电脑上储存在一个文件中
 - 能够跟踪长期信息
 - 潜在安全隐患, 因为用户(或者他们运行的程序)能够打开 cookie 文件, 并查看/ 修改 cookie 的值, 等

Cookie 在电脑的什么地方?

- IE : HomeDirectory\ Cookies
 - 例如C:\ Documents and Settings \ administrator \ Cookies
 - 每一个保存为.txt 文件, 与网站的域名相似
- Firefox: %APPDATA%\Mozilla\ Firefox\ ??? .default \ cookies.txt (cookies.sqlite)
 - 查看Firefox中的cookie: Privacy, Show Cookies...



2、Caching 缓存

浏览器可以缓存 Web 页面以减少下载流量

- 为当前访问的页面维持一个临时的存储 (Cache)

1. 当服务器传输一个文档给浏览器时，会同时通过字段告诉浏览器文档最后修改时间

Last-Modified: Fri, 09 May 2008 16:42:06 GMT

浏览器将页面与头部字段同时存储在 Cache 中

2. 当浏览器再次浏览相同页面时，先检查它是否已经在浏览器的 Cache 中

Caching 缓存

3. 浏览器浏览页面时，如果发现不在 Cache 中
 - 发起常规的 GET 请求，当响应报文到达时，显示页面，并将其放在 Cache 中（连同头部信息）
4. 如果发现已经存储在 cache 中
 - 那么浏览器再次发送 GET 请求，并附上 Cache 中的文件日期，由服务器判断 Cache 中的内容是否过期
 - **If-Modified-Since:** Fri, 09 May 2008 16:42:06 GMT
5. 当服务器的响应报文到达时
 - 如果状态码是 200, 说明有更新, 将新版本存进 cache
 - 如果状态码是 304, 直接显示 cache 中的页面

Cache 的调试与查看

- 在 Firefox 中用 about:cache 作为地址，可以查看 Cache 中文件的状态

Information about the Network Cache Storage Service

☐ Private ☐ Anonymous AppID ☐ In Browser Element [Back to overview](#)

appcache

Number of entries: 549
Maximum storage size: 512000 KiB
Storage in use: 3390 KiB
Storage disk location: C:\Users\cheny\AppData\Local\Mozilla\Firefox\Profiles\gsdmjhob.default\OfflineCache

Key	Data size	Fetch count	Last Modified	Expires	Pinning
https://www.zybuluo.com/static/zybuluo.mdeditor.appcache	32422 bytes	1	2017-11-20 20:54:45	Expired Immediately	
https://www.zybuluo.com/static/img/favicon.png	787 bytes	1	2017-11-20 20:54:46	2018-11-20 17:13:03	
https://www.zybuluo.com/static/assets/zh-CN/zybuluo.com/zybuluo.mdeditor.appcache	28343 bytes	1	2017-11-20 20:54:46	2018-11-20 17:13:03	

3、Gzip 内容压缩

- HTTP/1.1 开始，服务器端和客户端可以共同使用压缩算法对传输的内容进行压缩，节省流量

- 客户端发送头部，声明支持的算法：

Accept-Encoding: gzip, deflate

- 服务器端用客户端支持的压缩算法压缩文件内容后再传输给客户端，并附上头部：

Vary: Accept-Encoding

Content-Encoding: gzip

GET	200	1.22 K	text/plain	http://www.inspector.com/autou...
K, ElapsedTime=14.438 s)				
s	GET	200	1.38 K	text/html http://sw.fzu.edu.cn/
s	GET	200	7.54 K	text/css http://sites/sw.fzu.edu.cn/files/c...

Status Code Definition

1380 bytes received by 192.168.1.2:4491

```

1 HTTP/1.1 200 OK
2 Date: Sun, 18 May 2008 11:01:55 GMT
3 Server: Apache/2.2.8 (Ubuntu) PHP/5.2.4-2ubuntu5.1 with Suhosin-Patch
4 X-Powered-By: PHP/5.2.4-2ubuntu5.1
5 Expires: Sun, 19 Nov 1978 05:00:00 GMT
6 Last-Modified: Sun, 18 May 2008 11:01:55 GMT
7 Cache-Control: store, no-cache, must-revalidate
8 Cache-Control: post-check=0, pre-check=0
9 Vary: Accept-Encoding
10 Content-Encoding: gzip
11 Content-Length: 10382
12 Keep-Alive: timeout=15, max=100
13 Connection: Keep-Alive
14 Content-Type: text/html; charset=utf-8

```

看过去像乱码的报文内容，
实为经过压缩的数据

```

15
16 ?
17 R船口
18 ? 蠕 hnd卵
19
20 蒜8旬k耀€魔#阪0皂B? 錄默60z ? \7+透\PB 嚙b綽? P 蕓00縲? 狝? '賊時G樽啍? x? , #>ㄣy? 蟲底dc1N礪00 硯0
21 尤#=蛋0翊磨=60
22 ? 瞞妊0\坂? 2滯榔 ? 'V鐘0? 倖Fq耗#腰0騁悵為? R經情08鑿8,<桃郵10 貪 00堞厨礪q0? 睥醋r0慄撿<莢q

```

4、ETag 实体标记

- ETag 是一个可以与 Web 资源关联的记号 (特殊的字符串), 由服务器生成, 可用来判断一个资源是否有被修改
- 服务器端响应报文中加入 ETag:

HTTP/1.1 200 OK

Last-Modified: Tue, 12 Dec 2006 03:03:59 GMT

ETag: "10c24bc-4ab-457e1c1f"

Content-Length: 12195

ETag

- 客户端再次请求这个资源时:

GET /i/yahoo.gif HTTP/1.1

Host: us.yimg.com

If-Modified-Since: Tue, 12 Dec 2006 03:03:59 GMT

If-None-Match: "10c24bc-4ab-457e1c1f"

- 服务器端响应

HTTP/1.1 304 Not Modified

ETag

- ETag 的作用与 Cache/Last-Modified 有点类似
- 如果 ETag 匹配，则服务器返回 304 代码，从而节省了下载时间
- ETag 的问题在于它们的构造算法不唯一，每个服务器都不相同，不同服务器之间不能共用 ETag

5、内容协商

- 功能

字段

- 媒体类型协商

Accept

- 语言协商

Accept-Language

- 字符集协商

Accept-Charset

- 编码协商

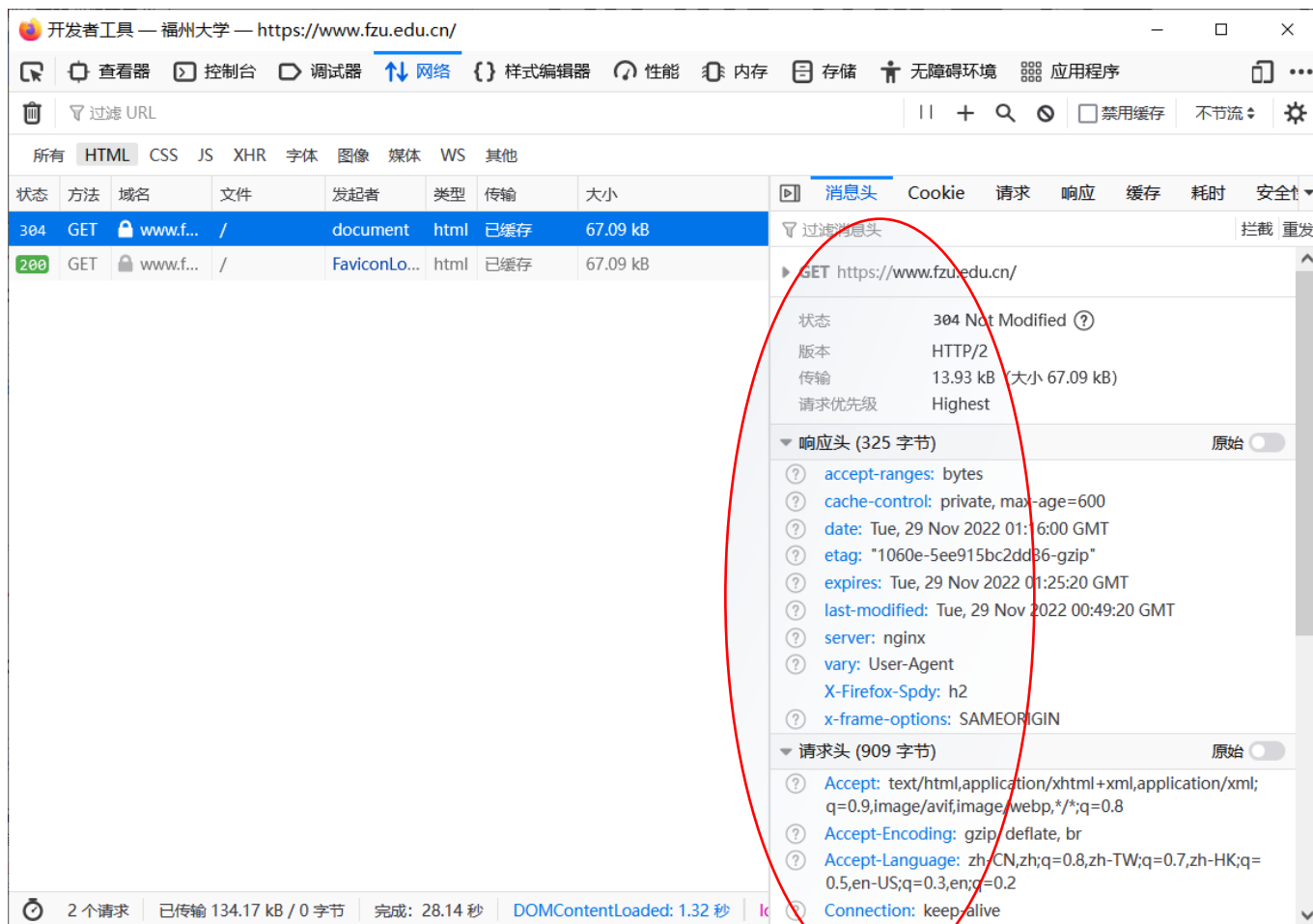
Accept-Encoding

6、身份认证

- WWW-Authentication Authorization
 - 用于表明被请求的资源需要登录



Firefox 开发者工具——网络



Things to do

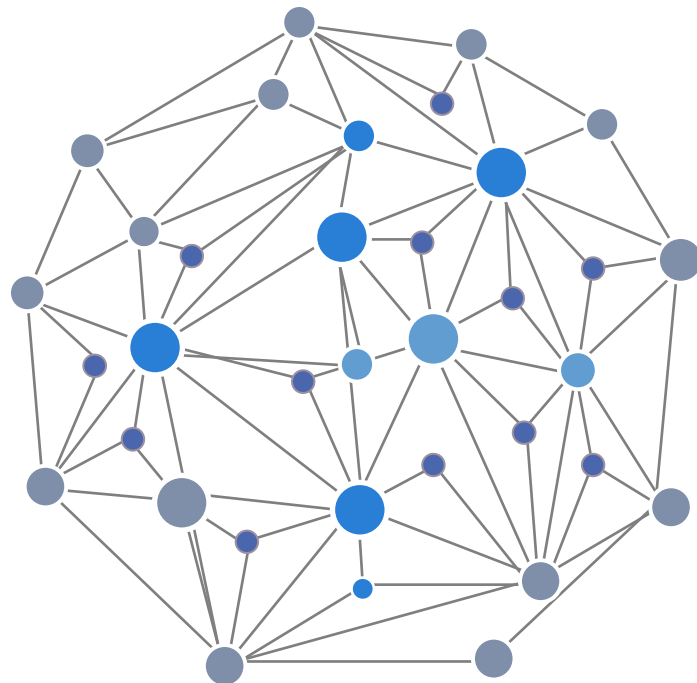
- 学习
 - <https://how2j.cn/k/http/http-tutorials/568.html>
- 作业：
 - 完成一个简单的 HTTP 服务器
 - <http://chenyv.gitee.io/webprogramming/homeworks/webserver/>

THANKS

本章结束

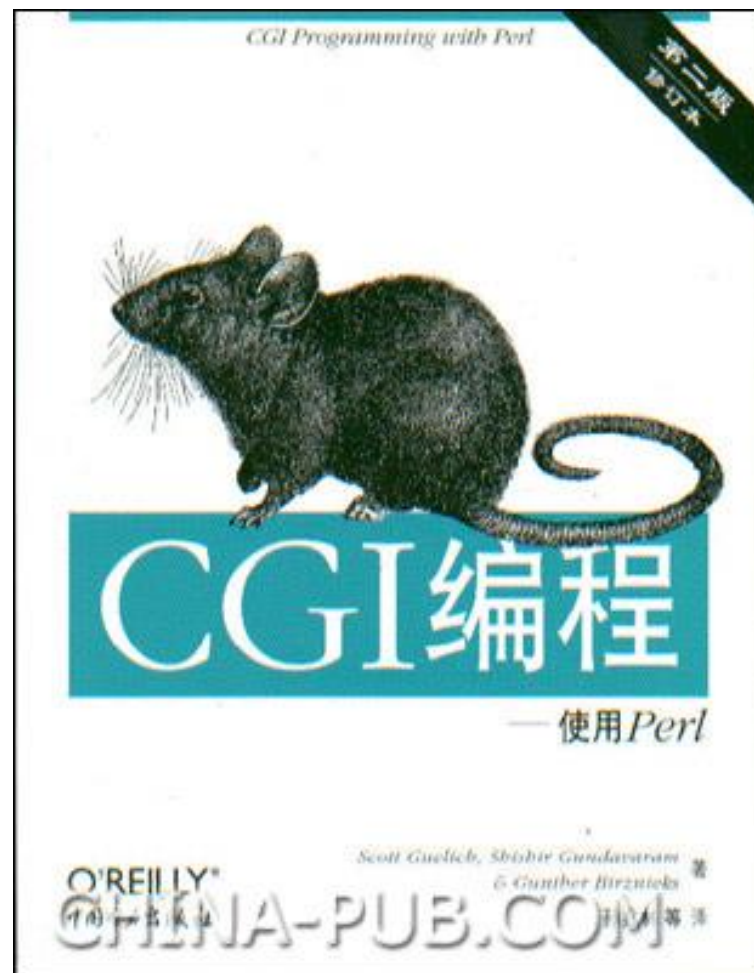
陈昱

福州大学 计算机与大数据学院 软件工程系

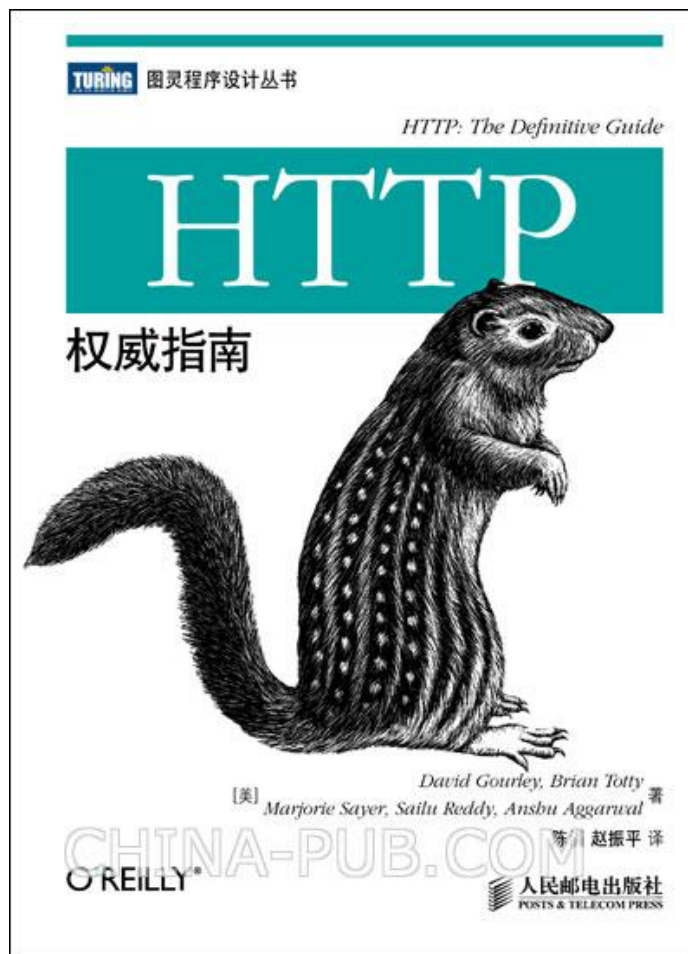


Extended Reading

- O'Reilly
- CGI 编程 - 使用 Perl (第二版)
- 第2章 HTTP
- 比较详细



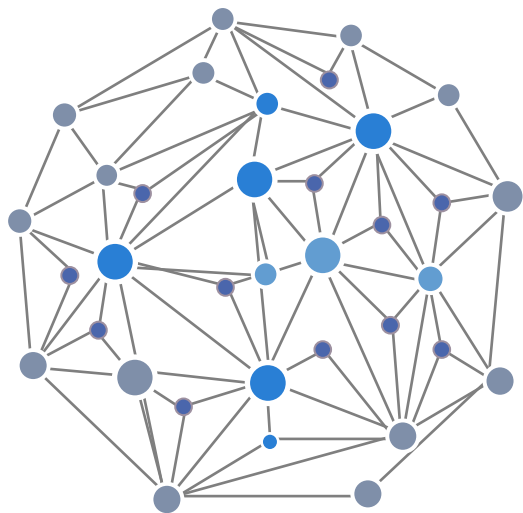
Extended Reading



- O'Reilly Media
 - HTTP权威指南
 - 最新出版，比较厚
-
- 关于 HTTP, 其他关于网络原理的书籍都有简要介绍

Reference

- Hypertext Transfer Protocol -- HTTP/1.1
 - <http://www.w3.org/Protocols/rfc2616/rfc2616.html>
- PERSISTENT CLIENT STATE HTTP COOKIES
 - http://wp.netscape.com/newsref/std/cookie_spec.html
- about: URI scheme
 - http://en.wikipedia.org/wiki/About:_URI_scheme



Web 性能优化

网页加速的35条优化法则 by Yahoo!

- 原为 14 条，后来加了 21 条新法则
 - **Best Practices for Speeding Up Your Web Site**
 - <http://developer.yahoo.com/performance/rules.html>
- 14 条的中文翻译
 - http://blog.sina.com.cn/s/blog_4fb2c3f001000a4t.html

Extended Reading

14 Steps to Faster-Loading Web Sites

High Performance Web Sites

Essential Knowledge
for Front-End Engineers



O'REILLY®

High Performance Web Sites
14 Steps to Faster-Loading Web Sites
Steve Souders 作者推荐

高性能网站建设指南

前端工程师技能精髓



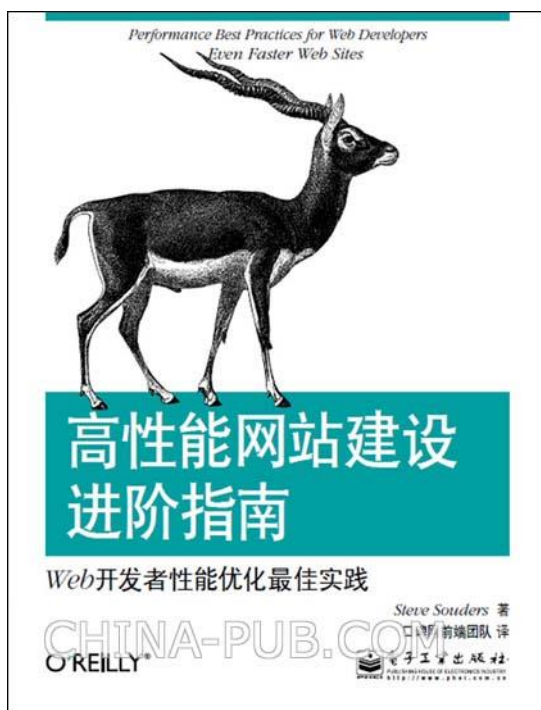
Steve Souders 著
刘松博 译

O'REILLY®

电子工业出版社
www.eip.com.cn

- O'Reilly
- High Performance Web Sites
- Steve Souders
- 中文版
 - 高性能网站建设指南
 - <http://book.douban.com/subject/3132277/>

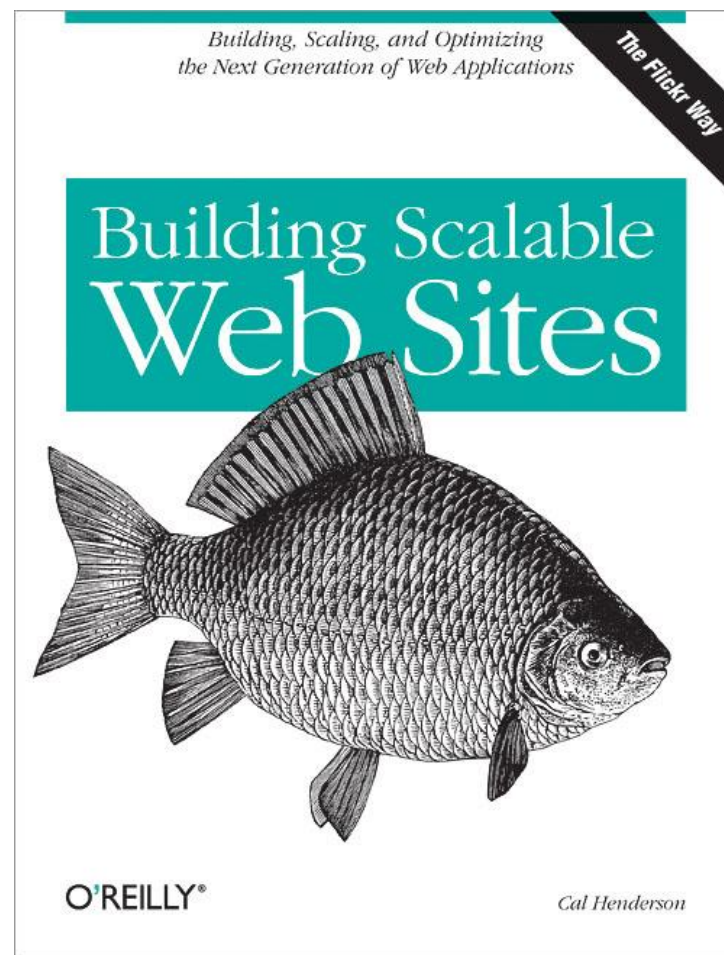
Extended Reading

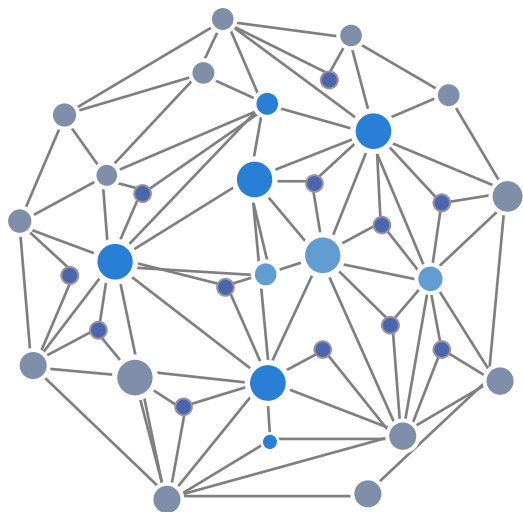


- O'Reilly
- 高性能网站建设进阶指南：Web开发者性能优化最佳实践
- 前一本书的补充

Extended Reading

- O'Reilly
- Building Scalable Web Sites
- Yahoo!
- 中文版
 - 构建可扩展的Web站点





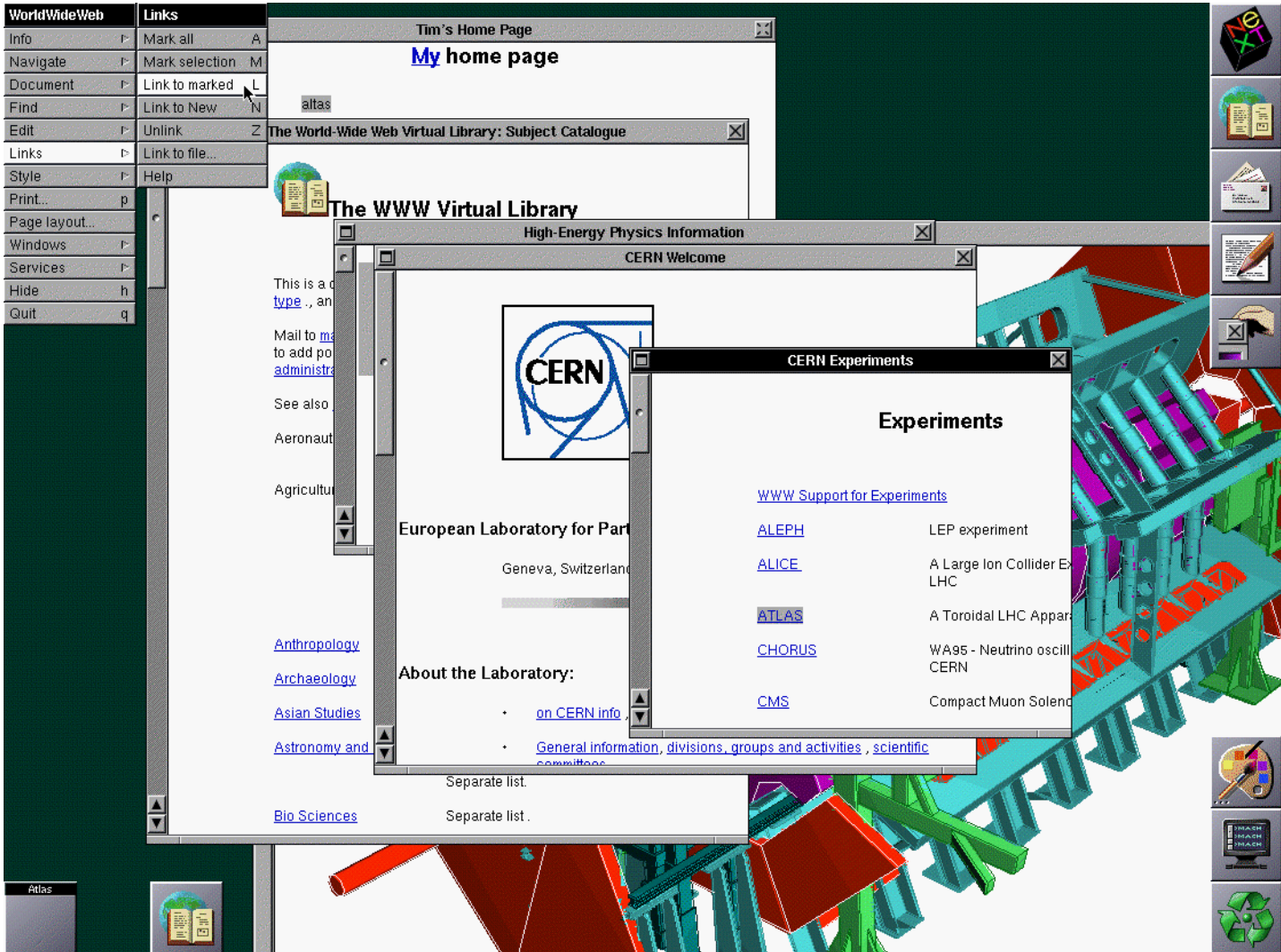
Web 浏览器

Web Client: Browser

Web 的客户端：浏览器

- 与服务器进行交互，获取并显示数据
- 从文本向图形界面, 从桌面向移动设备发展
- “已成为世界上最常用的软件”

世界上第一个 Web 浏览器: WorldWideWeb (Tim Berners-Lee)



浏览器的渲染引擎

- 渲染引擎，或称排版引擎
 - Trident: Internet Explorer 所使用
 - Gecko: Netscape 6 以后版本以及 Firefox 所使用
 - KHTML: KDE 开发团队所开发，主要由 Konqueror 所使用
 - WebKit: Apple 修改 KHTML 而来，主要由 Safari 所使用，还有 Google Android, N S60
 - Presto: Opera 7 开始采用

支持标准

- HTTP（超文本传输协议）和 HTTPS
- 无线应用协议 Wireless Application Protocol, WAP
- HTML, XHTML 及 XML
- 图形档案格式如 GIF、PNG、JPEG、SVG
- CSS
- JavaScript
- Cookie
- Adobe Flash
- Java applet
- Favicons
- 数字签名证书

基本功能

- 书签管理
- 下载管理
- 网页内容缓存 (Cache)
- 通过第三方插件 (plugins) 支持多媒体

附加功能

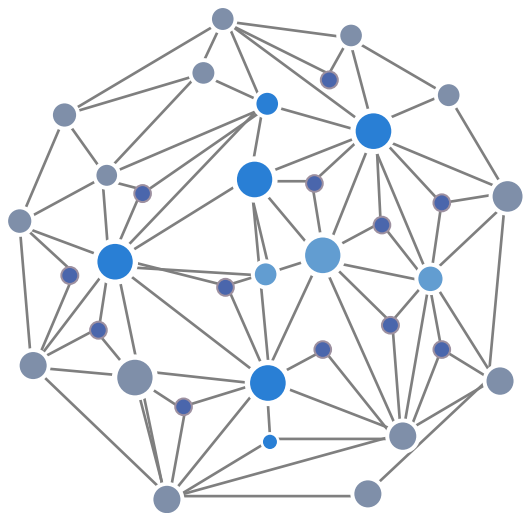
- 网址和表单资料自动完成
- 分页浏览
- 禁止弹出式广告
- 广告过滤
- 反钓鱼(伪造网址) Anti-Phishing

手机浏览器

- **Android** (mobile phone platform)
 - by Google (based on WebKit)
- **Internet Explorer Mobile**
 - by Microsoft Inc
- **Minimo**
 - by Mozilla Foundation (based on Gecko)
- **Opera Mobile**
 - by Opera Software ASA
- **Safari**
 - by Apple Inc on iPhone (based on WebKit)
- **Web Browser for S60**
 - by Nokia (based on WebKit)

阅读材料

- 《程序员》2008年 第 3 期特别报道
 - 用户为本：浏览器的生存之道
 - 浏览器：向 Web 标准看齐
 - 山雨欲来：手机浏览器



Web 服务器软件

Web Server

Web Server

- 一般把提供 HTTP 服务的服务器为 Web Server, 提供 http:80 (https: 443) 服务
- 通常运行在OS的后台, 对浏览器请求提供响应
- 目前流行的 Web Server 基本都是从下面两个软件衍生来的
 - CERN
 - NCSA

流行的 Web Server 软件

- 重量级
 - Apache (Apache Software Foundation)
 - IIS (Microsoft)
- 轻量级
 - Nginx
 - lighttpd
 - Litespeed
 -

Apache

- Apache
 - 开源 open source
 - 快速 fast
 - 稳定 reliable
 - 可运行于各种操作系统 portable
 - 占有率世界第一

Apache Resources

- 官方网站:
 - <http://httpd.apache.org/>
- Apache HTTP Server Version 2.2 中文文档
 - <http://lamp.linux.gov.cn/Apache/ApacheMenu/index.html>
 - http://sw.fzu.edu.cn/WebProgramming/ApacheManual_zh-CN/
- Apache 非官方 Win32 二进制文件下载
- 提供了 SSL 等模块
 - <http://www.apachelounge.com/download/>

IIS (Internet Information Services)

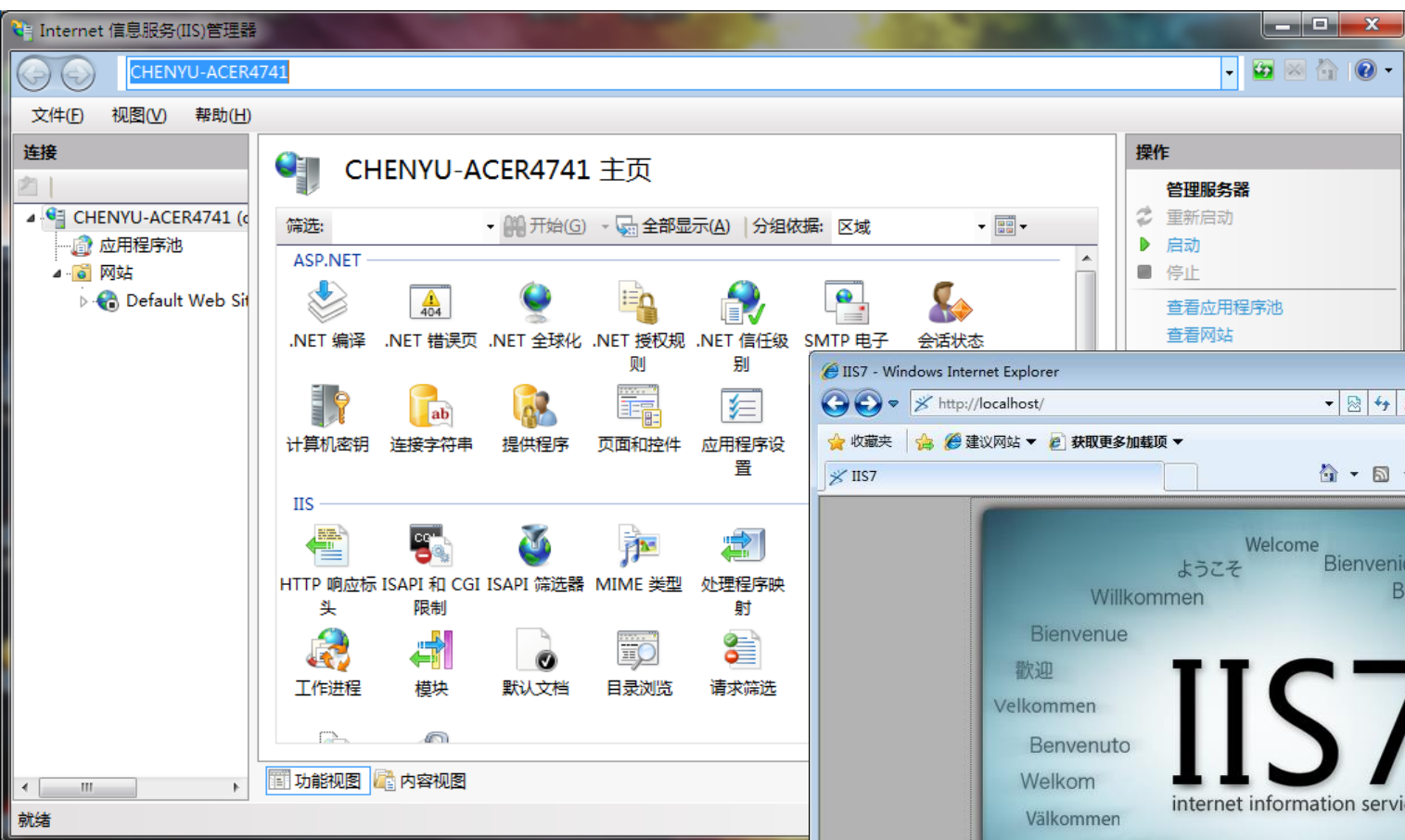
- Microsoft Windows IIS
 - 是一个基于 Internet 的服务器集合
 - HTTP, FTP, SMTP, ...
 - 通过 GUI 操作与维护
 - 只能运行在 Windows 下
 - 占有率世界第二

IIS 5.1 in WinXP

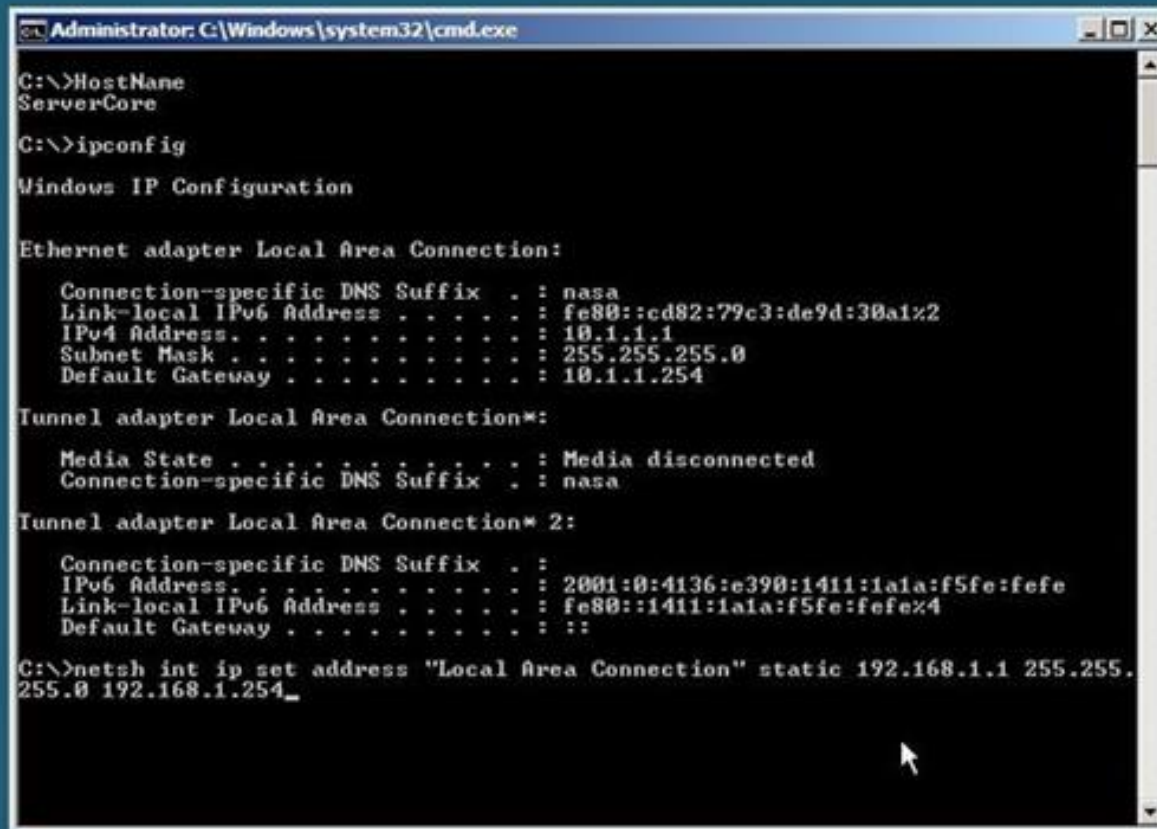


在 C:\Inetpub\wwwroot 中创建一个 hello.html 文件
访问 <http://localhost/hello.html> 或 http://your_ip/hello.html

IIS 7/7.5 in Vista/Win7



Windows Server 2008 Server Core



```
Administrator: C:\Windows\system32\cmd.exe

C:\>hostname
ServerCore

C:\>ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : nasa
    Link-local IPv6 Address . . . . . : fe80::cd82:79c3:de9d:30a1%2
    IPv4 Address. . . . . : 10.1.1.1
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 10.1.1.254

Tunnel adapter Local Area Connection*:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : nasa

Tunnel adapter Local Area Connection* 2:

    Connection-specific DNS Suffix  . : 
    IPv6 Address. . . . . : 2001:0:4136:e398:1411:1a1a:f5fe:fefe
    Link-local IPv6 Address . . . . . : fe80::1411:1a1a:f5fe:fefex4
    Default Gateway . . . . . : ::

C:\>netsh int ip set address "Local Area Connection" static 192.168.1.1 255.255.
255.0 192.168.1.254_
```

Windows Server (R) 2008 Enterprise
Evaluation copy, Build 6001

常见 Linux 服务器发行版

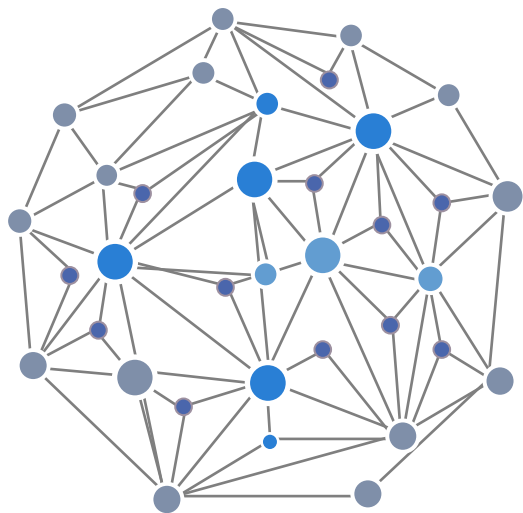
- RedHat Enterprise Linux (RHEL)
 - 收费
 - CentOS (RHEL 的开源社区自行编译版, 推荐)
- SUSE Linux Enterprise
 - 收费
- Debian / Ubuntu Server Edition
 - 免费, 推荐
- Linux 服务器的安装配置教程
 - <http://www.howtoforge.org/>

常见 Windows 服务器版

- Windows 2000 Server
- Windows Server 2003
- Windows Server 2008 (R2)
- Windows Server 2012 (R2)

Apache 配置手册

- Apache HTTP Server Version 2.2 中文文档
 - <http://lamp.linux.gov.cn/Apache/ApacheMenu/index.html>
 - http://sw.fzu.edu.cn/WebProgramming/ApacheManual_zh-CN/



服务器端程序原理 与通用网关接口 CGI

服务器端程序到底做了什么？

- 对客户端（浏览器）来说，服务器端程序发送过来的是什么？
- 发送过来的就是客户端的代码，包括HTML, CSS, JavaScript, 图片等
- 服务器端程序所作的事情就是生成客户端所能浏览的代码（从底层来说）

“动态网站”

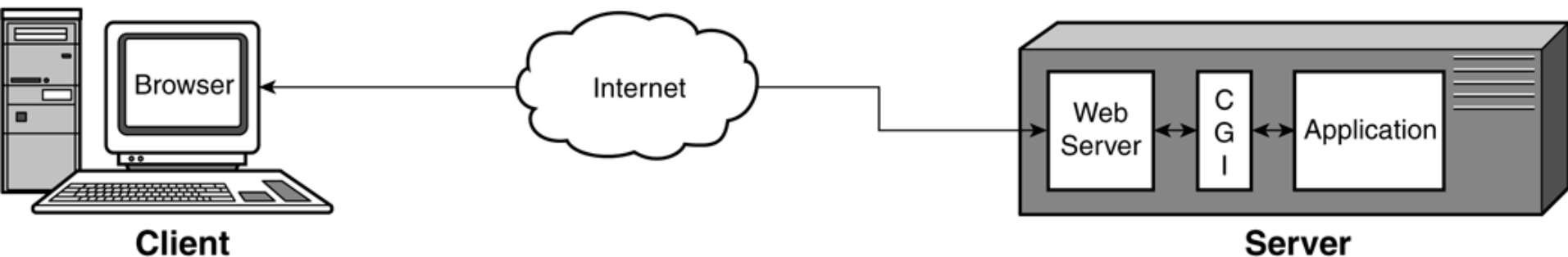
- 动态网站指的是网站内容是在浏览器访问Web服务器时才由应用程序动态创建
- 动态网页和静态网页之间的主要差别体现在服务器一端。这主要是文档内容的生成方法不同
- 这主要是文档内容的生成方法不同，而从浏览器的角度看，这两种文档并没有区别！

什么是 CGI?

- CGI 是一种标准
 - 通用网关接口 (Common Gateway Interface)
- 它定义了**动态网页**应如何构建，输入数据应如何提供给应用程序，以及输出结果应如何使用
- CGI 程序就是可以根据用户输入生成动态页面的程序（可以使用多种编程语言实现，C/C++，Perl，... 甚至VB, Delphi)

CGI 原理

- CGI 程序是可执行文件，由 Web 服务器通过 CGI(通用网关接口) 调用
- CGI 程序的输出（即客户端代码）被 Web 服务器通过 Internet 发送给用户的浏览器



CGI 的运行

- 在 Apache 服务器下，CGI 程序一般位于 DocumentRoot 下的 cgi-bin 子目录里
- 可以用任何语言编写，最常用的是 Perl (.pl)
- 当然也可以是 C 语言，下面是一个例子

hello.c

```
#include <stdio.h>
#include <stdlib.h>
```

```
int main() {
    // http 头部信息
```

```
    printf("Content-Type:text/html \n\n");
```

```
    printf(" <html> \n");
```

```
    printf(" <head> \n");
```

```
    printf(" <title> Result</title> \n");
```

```
    printf(" </head> <body> \n");
```

```
    printf("\n<p> Hello,World</p> \n");
```

```
    printf(" </body> </html> \n");
```

```
    return 0;
```

```
}
```

```
gcc hello.c -o hello.cgi
chmod 755 hello.cgi
本质上是一个I/O重定向
```

CGI 原理

- CGI 程序通常通过标准输出流 (stdout) 创建 HTTP 响应内容
 - 比如 C 的 `printf`, C++ 的 `cout`
- 通常还必须输出头部的 `content-type` 字段
 - **`Content-type: text/html`**
 - 这是头部的最后一行, 因此后跟一个空行
- 响应中剩下的部分用来打印 HTML 代码到标准输出

用 CGI 处理表单数据

- CGI 程序最常见的应用就是表单处理
- 浏览器将用户填写的表单数据通过HTTP发送给 CGI 程序
- CGI 程序根据用户提交的数据，计算得到返回结果

CGI 程序获取表单数据

- 当使用 GET 方法时，查询串被放在 OS 的环境变量 QUERY_STRING 中
 - 可通过 getenv 等系统调用得到
- 当使用 POST 方法时，表单数据的查询串可以从标准输入 (stdin) 得到
 - 比如 C 的 scanf, C++ 的 cin
 - 环境变量 CONTENT_LENGTH 告诉程序需要读取多少个字节

http://...../form.html

```
<form action="cgi-bin/hello.cgi" method="get">
  <p>Enter your message:
  <input type="text" name="msg" maxlength="200" />
  <input type="Submit" value="提交"/>
  <input type="Reset" name="clear" value="Reset" />
  </p>
</form>
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
int main() {
```

```
    char* query;
```

```
    // http 头部信息
```

```
    printf("Content-Type:text/html;  
           charset=utf-8\n\n");
```

```
    printf("<html>\n");
```

```
    printf("<head>\n");
```

```
    printf("<title>Result</title>\n");
```

```
    printf("</head><body>\n");
```

```
    query = getenv("QUERY_STRING");
```

```
    // 在这里可以对表单数据进行复杂的处理
```

```
    printf("\n<p> %s </p> </body> </html>\n", query);
```

```
    return 0;
```

```
}
```

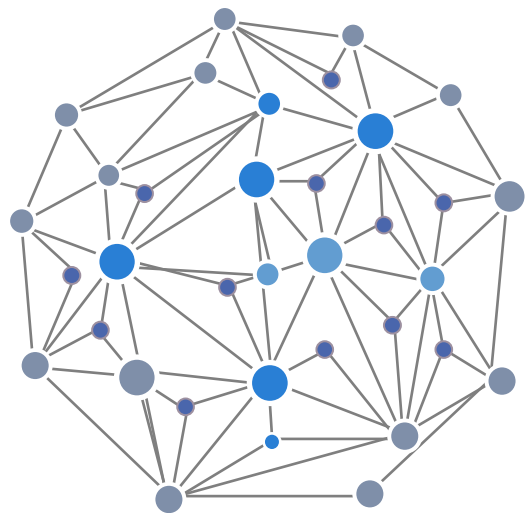
gcc -o hello.cgi hello.c

CGI 的缺点

- 执行速度较慢
 - Web 服务器每收到一个请求，就必须创建一个新的 CGI 进程，相对服务器资源代价比较高
- 安全性
 - 如果程序权限设置不当，CGI 会威胁系统安全
- 开发速度
 - 服务器端性能的瓶颈在于I/O，因此用C++写的 CGI 速度不会比 PHP, JSP 这些语言快非常多
 - 选择服务器端语言很大程度上取决于**开发速度**

CGI 的发展

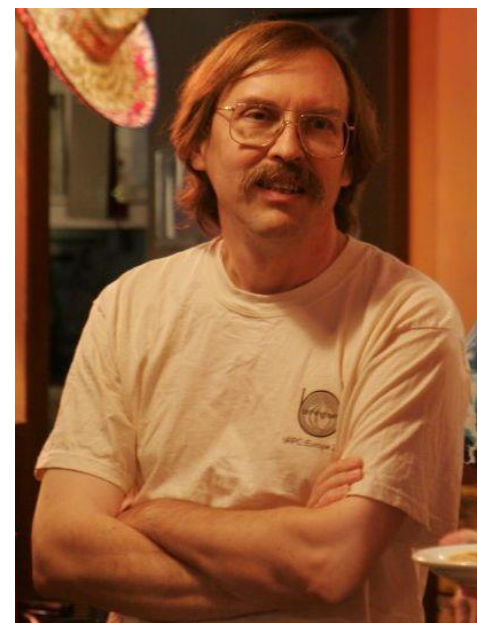
- 早期的 CGI 程序存在上述缺点，因此逐渐退出了主流
- 不过 CGI 也在发展，比如：
 - fast-cgi
 - mod_perl, mod_python, mod_ruby
- 此外 CGI 的优势在于可以用任何语言编写，比 Java, .NET 这些平台来得自由



Perl 语言简介

Perl 语言

- 发明者, Larry Wall, 1987年
- Perl 是一种自由且功能强大的编程语言
- 应用广泛
 - 系统管理 (网管的必备工具)
 - CGI 编程
 - 文字数据处理
 - 生物数据处理 (DNA 序列分析, 生物学家的好帮手, 在人类基因组计划中做出贡献)



Perl 的特点

- Perl 是一种解释型语言
- Perl 语言最强大的地方是它的正则表达式处理
 - 不过有 PCRE 函数库后, C 语言也能使用和 Perl 一样的正则表达式
- 另外还有一个网上庞大代码库 CPAN

Perl 的运行

- 运行 Perl 程序需要安装 Perl 解释器
- Linux 发行版一般都带有 Perl
 - 没有的话也可以在线安装
- Windows 下最常用的是 ActivePerl 套装
- 命令行执行: perl 程序名.pl
 - 后面可跟参数

```
# word_table.pl
```

```
#>>> Main loop to get and process lines of input text
```

```
while (<>) {
```

```
#>>> Split the line into words
```

```
    @line_words = split /[ \.,:!\?]\s*/;
```

```
#>>> Loop to count the words (either increment or initialize to 1)
```

```
    foreach $word (@line_words) {
```

```
        if (exists $freq{$word}) {
```

```
            $freq{$word}++;
```

```
        } else {
```

```
            $freq{$word} = 1;
```

```
        }
```

```
    }
```

```
}
```

```
#>>> Display the words and their frequencies
```

```
print "\n Word \t\t Frequency \n\n";
```

```
foreach $word (sort keys %freq) {
```

```
    print " $word \t\t $freq{$word} \n";
```

```
}
```

Perl CGI 的例子：最上面的一行称为 "shebang"，用于指定 Perl 解释器的路径

```
#!/usr/bin/perl -w
```

```
# reply.cgi
```

```
# This CGI program returns a greeting to the client
```

```
print "Content-type: text/html \n\n",  
      "<?xml version = '1.0'?> \n",  
      "<!DOCTYPE html PUBLIC '-//w3c//DTD XHTML 1.1//EN'\n",  
      "'http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd'>\n",  
      "<html xmlns = 'http://www.w3.org/1999/xhtml'>\n",  
      "<head> <title> reply.cgi example </title> </head>\n",  
      "<body>\n",  
      "<h1> Greetings from your Web server! </h1>\n",  
      "</body> </html>";
```

Perl 相关资源

- <http://perlchina.org/>
 - 中国 Perl 社区
- <http://cpan.org/>
 - Perl 开源模块库, 7000个现成模块可供使用
- <http://www.perl.org>
 - 官方网站
- ActiveState Perl (Windows 下推荐)
 - www.activestate.com/Products/activeperl

Extended Reading

- O'Reilly Perl 系列
- Perl 语言入门(第四版)
 - 小骆驼书
- Perl 语言编程(第3版)
 - 大骆驼书
- 高级 Perl 编程
 - 黑豹书

