第1章　HTTP 概述

1.1　HTTP——因特网的多媒体信使

- HTTP 使用的是可靠的数据传输协议　　因此即使数据来自地球的另一端，它也能够确保数据在传输的过程中不会被损坏或产生混乱

1.2　Web 客户端和服务器

- Web 内容都是存储在 Web 服务器上的
- Web 服务器所使用的是 HTTP 协议，因此经常会被称为 HTTP 服务器
- HTTP 服务器存储了因特网中的数据，如果 HTTP 客户端发出请求的话，它们会提供数据

1.3　资源

- Web 服务器是 Web 资源（Web resource）的宿主
- Web 资源是 Web 内容的源头
- 最简单的 Web 资源就是 Web 服务器文件系统中的静态文件
- 但资源不一定非得是静态文件。资源还可以是根据需要生成内容的软件程序
- 所有类型的内容来源都是资源

1.3.1　媒体类型

- HTTP 给每种要通过 Web 传输的对象都打上了名为 **MIME 类型（MIME type）**的数据格式标签
- MIME 类型是一种文本标记，表示一种主要的对象类型和一个特定的子类型，中间由一条斜杠来分隔
  - 常见的 MIME 类型有数百个
  - HTML 格式的文本文档由 text/html 类型来标记
  - 普通的 ASCII 文本文档由 text/plain 类型来标记
  - JPEG 格式的图片为 image/jpeg 类型
- Web 服务器会为所有 HTTP 对象数据附加一个 MIME 类型
- 当 Web 浏览器从服务器中取回一个对象时，会去查看相关的 MIME 类型，看看它是否知道应该如何处理这个对象

1.3.2.　URI

- 服务器资源名被称为**统一资源标识符**（Uniform Resource Identifier，URI）
  - URL
  - URN

1.3.3.　URL

- **统一资源定位符（URL）**是资源标识符最常见的形式
- URL 描述了一台特定服务器上某资源的特定位置　　可以明确说明如何从一个精确、固定的位置获取资源
- 大部分 URL 都遵循一种标准格式，这种格式包含三个部分
  - URL 的第一部分被称为**方案**（scheme），说明了访问资源所使用的协议类型
  - 第二部分给出了服务器的因特网地址（比如，www.joes-hardware.com）
  - 其余部分指定了 Web 服务器上的某个资源（比如，/specials/saw-blade.gif）

1.3.4.　URN

- **统一资源名**URN 是作为特定内容的唯一名称使用的，与目前的资源所在地无关
- 使用这些与位置无关的 URN，就可以将资源四处搬移
- 通过 URN，还可以用同一个名字通过多种网络访问协议来访问资源
- URN 仍然处于试验阶段，还未大范围使用
- 除非特殊说明，否则本书的其余部分都会使用约定的术语，并且会不加区别地使用 URI 和 URL

1.4　事务

- 一个 HTTP 事务
  - 一条（从客户端发往服务器的）请求命令
  - 一个（从服务器发回客户端的）响应结果组成
  - 这种通信是通过名为 **HTTP 报文**（HTTP message）的格式化数据块进行的

1.4.1　方法

- HTTP 支持几种不同的请求命令，这些命令被称为 **HTTP 方法**（HTTP method）
- 每条 HTTP 请求报文都包含一个方法
- 这个方法会告诉服务器要执行什么动
- **五种常见的 HTTP 方法**
  - GET　　从服务器向客户端发送命名资源
  - PUT　　将来自客户端的数据存储到一个命名的服务器资源中去
  - DELETE　　从服务器中删除命名资源
  - POST　　将客户端数据发送到一个服务器网关应用程序
  - HEAD　　仅发送命名资源响应中的HTTP 首部

1.4.2　状态码

- 每条 HTTP 响应报文返回时都会携带一个状态码
- 状态码是一个三位数字的代码　　告知客户端请求是否成功，或者是否需要采取其他动作
- HTTP 还会发送一条解释性的“原因短语”文本　　包含文本短语主要是为了进行描述

1.4.3　Web页面中可以包含多个对象

- 应用程序完成一项任务时通常会发布多个 HTTP 事务

1.5　报文

- HTTP 报文是由一行一行的简单字符串组成的
- HTTP 报文都是纯文本
- 从 Web 客户端发往 Web 服务器的 HTTP 报文称为**请求报文**（request message）
- 从服务器发往客户端的报文称为**响应报文**（response message）

HTTP 报文包括以下三个部分

- 起始行　　报文的第一行就是起始行，在请求报文中用来说明要做些什么，在响应报文中说明出现了什么情况
- 首部字段　　起始行后面有零个或多个首部字段
  - 每个首部字段都包含一个名字和一个值
  - 为了便于解析，名与值之间用冒号（:）来分隔
  - 首部以一个空行结束
- 主体
  - 空行之后就是可选的报文主体了，其中包含了所有类型的数据
  - 请求主体中包括了要发送给 Web 服务器的数据
  - 响应主体中装载了要返回给客户端的数据
  - 体中可以包含任意的二进制数据（比如图片、视频、音轨、软件程序）

1.6　连接

1.6.1　TCP/IP

- HTTP 是个应用层协议
- 它把联网的细节都交给了通用、可靠的因特网传输协议 TCP/IP
- **TCP 提供了**
  - 无差错的数据传输
  - 按序传输（数据总是会按照发送的顺序到达）
  - 未分段的数据流（可以在任意时刻以任意尺寸将数据发送出去）
- 因特网自身就是基于 TCP/IP 的

1.6.2　连接、IP地址及端口号

- 在 HTTP 客户端向服务器发送报文之前，需要用**网际协议**（Internet Protocol，IP）地址和端口号在客户端和服务器之间建立一条 TCP/IP 连接
- 在 TCP 中，你需要知道服务器的 IP 地址，以及与服务器上运行的特定软件相关的 TCP 端口号
- **主机名**就是 IP 地址比较人性化的别称
- 可以通过一种称为**域名服务**（Domain Name Service，DNS）的机制方便地将主机名转换为 IP 地址

1.7　协议版本

1.8　Web 的结构组件

1.8.1　代理

- 代理位于客户端和服务器之间
- 接收所有客户端的 HTTP 请求，并将这些请求转发给服务器（可能会对请求进行修改之后转发）
- 对用户来说，这些应用程序就是一个代理，代表用户访问服务器。
- 出于安全考虑，通常会将代理作为转发所有 Web 流量的可信任中间节点使用
- 代理还可以对请求和响应进行过滤

1.8.2　缓存

- **Web 缓存**（Web cache）或**代理缓存**（proxy cache）是一种特殊的 HTTP 代理服务器
- 可以将经过代理传送的常用文档复制保存起来　　下一个请求同一文档的客户端就可以享受缓存的私有副本所提供的服务了

1.8.3　网关

- **网关**（gateway）是一种特殊的服务器，作为其他服务器的中间实体使用
- 通常用于将 HTTP 流量转换成其他的协议

1.8.4　隧道

- **隧道**（tunnel）是建立起来之后，就会在两条连接之间对原始数据进行盲转发的 HTTP 应用程序
- HTTP 隧道通常用来在一条或多条 HTTP 连接上转发非 HTTP 数据，转发时不会窥探数据

1.8.5.　Agent代理

- **用户 Agent 代理**（或者简称为 Agent 代理）是代表用户发起 HTTP 请求的客户端程序
- 所有发布 Web 请求的应用程序都是 HTTP Agent 代理