

02 | HTTP是什么? HTTP又不是什么?

Chrono 2019-05-31



显然，这个答案有点过于简单了，不能让他满意，他肯定会再追问你一些问题：

- HTTP 有什么特点? 有什么优点和缺点?



其实“协议”并不仅限于计算机世界，现实生活中也随处可见。例如，你在刚毕业时，会签一个“三方协议”，找房子时会签一个“租房协议”，公司入职时还可能会签一个

- 刚才说的这几个都是“协议”，本质上与 HTTP 是相同的。那么
- 第一点，协议必须要有两个或多个参与者，也就是“协”。
- 如果**只有**你一个人，那你自然可以想干什么就干什么，想怎么玩他人也不会干涉你，也就不需要所谓的“协议”。但是，一旦有保证最基本的顺畅交流，协议就自然而然地出现了。

利入住，“租房协议”里的参与者有两个：你和房东。

此以音叶著右各人分上者

做错了怎么办，有没有补救措施等等。例如，租房协议里就约定了，租期多少，租金多少，押金是多少，水电费谁来付，违约应如何处理等等。

HTTP 是一个用在计算机世界里的协议。它使用计算机能够理解的语言确立了一种计算机之间交流通信的规范，以及相关的各种控制和错误处理方式。

接下来我们看 HTTP 字面里的第二部分：“传输”。

同，例如广播协议、寻址协议、路由协议、隧道协议、选举协议等等。

也就是说，有两个最基本的参与者 A 和 B，从 A 开始到 B 结束，数据在 A 和 B 之间双向而不是

方 B。双方约定用 HTTP 协议来通信，于是浏览器把一些数据发给网站，网站再把一些数据发回给浏览器，最后展现在屏幕上，你就可以看到各种有意思的新闻、视频了。

传输, 就可以添加任意的额外功能, 例如安全认证、数据压缩、编

说到这里，你差不多应该能够明白 HTTP 的

讲了“协议”和“传输”，现在，我们终于到 HTTP 字面里的第三部分：**“超文本”**。

既然 HTTP 是一个“传输协议”，那么它传输的“超文本”到底是什么呢？我还是用两点来进一步解释。

所谓**“文本”**（Text），就表示 HTTP 传输的不是 TCP/UDP 这些底层协议里被切分的杂乱无章的二进制包（datagram），而是完整的、有意义的数据，可以被浏览器、服务器这样的上层应用程序处理。

所谓“**超文本**”，就是“超越了普通文本的文本”，它是文字、图片、音频和视频等的混合体，最关键的是含有“超链接”，能够从一个“超文本”跳跃到另一个“超文本”，形成复杂的非线性、网状的结构关系。

对于“超文本”，我们最熟悉的就应该是 HTML 了，它本身只是纯文字文件，但内部用很多标签定义了对图片、音频、视频等的链接，再经过浏览器的解释，呈现在我们面前的就是一个含有多种视听信息的页面。

OK，经过了对 HTTP 里这三个名词的详细解释，下次当你再面对面试官时，就可以给出比“超文本传输协议”这七个字更准确更有技术含量的答案：“**HTTP 是一个在计算机世界里专门在两点之间传输文字、图片、音频、视频等超文本数据的约定和规范**”。

现在你对“HTTP 是什么？”应该有了比较清晰的认识，紧接着的问题就是“HTTP 不是什
么？”，等价的问题是“HTTP 不能干什么？”。想想看，你能回答出来吗？

因为 HTTP 是一个协议，是一种计算机间通信的规范，所以它**不存在“单独的实体”**。它不是浏
览器、手机 APP 那样的应用程序，也不是 Windows、Linux 那样的操作系统，更不是
Apache、Nginx、Tomcat 那样的 Web 服务器。

且是一种“动态的存在”，是发生在网络连接、传输超文本数据时的一个“动态过程”。

HTTP 不是互联网。

互联网（Internet）是遍布于全球的许多网络互相连接而形成的一个巨大的国际网络，在它上面存放着各式各样的资源，也对应着各式各样的协议，例如超文本资源使用 HTTP，普通文件使用 FTP，电子邮件使用 SMTP 和 POP3 等。

毫无疑问，HTTP 是构建互联网的一块重要拼图，而且是占比最大的那一块。

HTTP 不是编程语言。

编程语言是人与计算机沟通交流所使用的语言，而 HTTP 是计算机与计算机沟通交流的语言，我们无法使用 HTTP 来编程，但可以反过来，用编程语言去实现 HTTP，告诉计算机如何用 HTTP 来与外界通信。

很多流行的编程语言都支持编写 HTTP 相关的服务或应用，例如使用 Java 在 T Web 服务，使用 PHP 在后端实现页面模板渲染，使用 JavaScript 在前端实现。

HTTP 不是 HTML, 这

HTML 是超文本的载体，是一种标记语言，使用各种标签描述文字、图片、超链接等资源，并且可以嵌入 CSS、JavaScript 等技术实现复杂的动态效果。单论次数，在互联网上 HTTP 传输最多的可能就是 HTML，但要是论数据量，HTML 可能要往后排了，图片、音频、视频这些类型的资源显然更大。

HTTP 不是一个孤立的协议。

在互联网世界里，HTTP 通常跑在 TCP/IP 协议栈之上，依靠 IP 协议实现寻址和路由、TCP 协议实现可靠数据传输、DNS 协议实现域名查找、SSL/TLS 协议实现安全通信。此外，还有一些协议依赖于 HTTP，例如 WebSocket、HTTPDNS 等。这些协议相互交织，构成了一个协议网，而 HTTP 则处于中心地位。

1. HTTP 是一个用在计算机世界里的协议，它确立了一种计算机之间交流通信的规范，以及相关
的各种控制和错误处理方式。
2. HTTP 专门用来在两点之间传输数据，不能用于广播、寻址或路由。
3. HTTP 传输的是文字、图片、音频、视频等超文本数据。
4. HTTP 是构建互联网的重要基础技术，它没有实体，依赖许多其他的技术来实现，但同时许多
技术也都依赖于它。

相关的**所有应用层技术的总和**”。

这里我画了一个思维导图，也可以算是这个专栏系列文章的“知识地图”。

```
graph LR; DNS((DNS)) --- OS1[操作系统]; DNS --- NetOS[网络OS]; DNS --- OS2[操作系统]; DNS --- LocalDNS[本地DNS];
```

IPv4

正向代理
反向代理

```

graph LR
    Browser[浏览器] --- ProtocolStack[协议栈]
    Browser --- WWW[方键网 WWW]
    ProtocolStack --- HTTP[HTTP]
    ProtocolStack --- URL[URL]
    HTTP --- RequestHeaderBody[请求头+请求体]
    HTTP --- ResponseHeaderBody[响应头+响应体]
    RequestHeaderBody --- RequestMethod[请求方法]
    RequestHeaderBody --- RequestResponse[请求/应答]
    RequestMethod --- GET[GET]
    RequestMethod --- HEAD[HEAD]
    RequestMethod --- POST[POST]
    RequestResponse --- RequestDocument[请求文档]
    RequestResponse --- RequestResponse2[请求/应答]
    RequestDocument --- RFC2616[RFC2616]
    RequestDocument --- RFC7230[RFC7230]
    RequestResponse2 --- RequestResponse3[请求/应答]
    RequestResponse2 --- RequestResponse4[请求/应答]
    URL --- ProtocolName[协议名]
    URL --- ResourceParameters[资源参数]
    URL --- Encoding[编码]
    WWW --- Chrome[Chrome]
    WWW --- Firefox[Firefox]
    WWW --- Safari[Safari]
    WWW --- IEEdge[IE/Edge]
    WWW --- Opera[Opera]
    HighPerformance[高性能] --- Nginx[Nginx]
    HighPerformance --- OpenResty[OpenResty]
  
```

```

graph LR
    HTTP[HTTP] --- Stateless[无状态]
    HTTP --- LongConn[长连接机制]
    HTTP --- Cache[缓存]
    HTTP --- MIME[MIME]
    HTTP --- CDN[CDN]
    HTTP --- Crawlers[爬虫]
    HTTP --- Proxies[代理]
    HTTP --- Auth[认证]
    HTTP --- Enc[加密]
    Enc --- HTTPS[HTTPS]
    Enc --- Sym[对称加密]
    Enc --- Asym[非对称加密]
    Enc --- Hash[摘要算法]
    Sym --- AES[AES]
    Sym --- ChaCha[ChaCha]
    Asym --- RSA[RSA]
    Asym --- ECC[ECC]
    Hash --- SHA1[SHA-1]
    Hash --- MD5[MD5]
    CDN --- LB[负载均衡]
    CDN --- Prox[就近访问]
    CDN --- Squid[Squid/Varnish/ATS]
    Crawlers --- Crawler[crawler/spider]
    Crawlers --- Robots[robots.txt]
    Proxies --- HTML4[HTML 4]
    Proxies --- HTML5[HTML 5]
    Auth --- Basic[Basic]
    Auth --- Digest[Digest]
    Auth --- OAuth[OAuth]
    Auth --- OAuth2[OAuth2]
    Auth --- OAuth3[OAuth3]
    Auth --- OAuth4[OAuth4]
    Auth --- OAuth5[OAuth5]
    Auth --- OAuth6[OAuth6]
    Auth --- OAuth7[OAuth7]
    Auth --- OAuth8[OAuth8]
    Auth --- OAuth9[OAuth9]
    Auth --- OAuth10[OAuth10]
    Auth --- OAuth11[OAuth11]
    Auth --- OAuth12[OAuth12]
    Auth --- OAuth13[OAuth13]
    Auth --- OAuth14[OAuth14]
    Auth --- OAuth15[OAuth15]
    Auth --- OAuth16[OAuth16]
    Auth --- OAuth17[OAuth17]
    Auth --- OAuth18[OAuth18]
    Auth --- OAuth19[OAuth19]
    Auth --- OAuth20[OAuth20]
    Auth --- OAuth21[OAuth21]
    Auth --- OAuth22[OAuth22]
    Auth --- OAuth23[OAuth23]
    Auth --- OAuth24[OAuth24]
    Auth --- OAuth25[OAuth25]
    Auth --- OAuth26[OAuth26]
    Auth --- OAuth27[OAuth27]
    Auth --- OAuth28[OAuth28]
    Auth --- OAuth29[OAuth29]
    Auth --- OAuth30[OAuth30]
    Auth --- OAuth31[OAuth31]
    Auth --- OAuth32[OAuth32]
    Auth --- OAuth33[OAuth33]
    Auth --- OAuth34[OAuth34]
    Auth --- OAuth35[OAuth35]
    Auth --- OAuth36[OAuth36]
    Auth --- OAuth37[OAuth37]
    Auth --- OAuth38[OAuth38]
    Auth --- OAuth39[OAuth39]
    Auth --- OAuth40[OAuth40]
    Auth --- OAuth41[OAuth41]
    Auth --- OAuth42[OAuth42]
    Auth --- OAuth43[OAuth43]
    Auth --- OAuth44[OAuth44]
    Auth --- OAuth45[OAuth45]
    Auth --- OAuth46[OAuth46]
    Auth --- OAuth47[OAuth47]
    Auth --- OAuth48[OAuth48]
    Auth --- OAuth49[OAuth49]
    Auth --- OAuth50[OAuth50]
    Auth --- OAuth51[OAuth51]
    Auth --- OAuth52[OAuth52]
    Auth --- OAuth53[OAuth53]
    Auth --- OAuth54[OAuth54]
    Auth --- OAuth55[OAuth55]
    Auth --- OAuth56[OAuth56]
    Auth --- OAuth57[OAuth57]
    Auth --- OAuth58[OAuth58]
    Auth --- OAuth59[OAuth59]
    Auth --- OAuth60[OAuth60]
    Auth --- OAuth61[OAuth61]
    Auth --- OAuth62[OAuth62]
    Auth --- OAuth63[OAuth63]
    Auth --- OAuth64[OAuth64]
    Auth --- OAuth65[OAuth65]
    Auth --- OAuth66[OAuth66]
    Auth --- OAuth67[OAuth67]
    Auth --- OAuth68[OAuth68]
    Auth --- OAuth69[OAuth69]
    Auth --- OAuth70[OAuth70]
    Auth --- OAuth71[OAuth71]
    Auth --- OAuth72[OAuth72]
    Auth --- OAuth73[OAuth73]
    Auth --- OAuth74[OAuth74]
    Auth --- OAuth75[OAuth75]
    Auth --- OAuth76[OAuth76]
    Auth --- OAuth77[OAuth77]
    Auth --- OAuth78[OAuth78]
    Auth --- OAuth79[OAuth79]
    Auth --- OAuth80[OAuth80]
    Auth --- OAuth81[OAuth81]
    Auth --- OAuth82[OAuth82]
    Auth --- OAuth83[OAuth83]
    Auth --- OAuth84[OAuth84]
    Auth --- OAuth85[OAuth85]
    Auth --- OAuth86[OAuth86]
    Auth --- OAuth87[OAuth87]
    Auth --- OAuth88[OAuth88]
    Auth --- OAuth89[OAuth89]
    Auth --- OAuth90[OAuth90]
    Auth --- OAuth91[OAuth91]
    Auth --- OAuth92[OAuth92]
    Auth --- OAuth93[OAuth93]
    Auth --- OAuth94[OAuth94]
    Auth --- OAuth95[OAuth95]
    Auth --- OAuth96[OAuth96]
    Auth --- OAuth97[OAuth97]
    Auth --- OAuth98[OAuth98]
    Auth --- OAuth99[OAuth99]
    Auth --- OAuth100[OAuth100]
    HTTPS --- Enc
    HTTPS --- Auth
  
```

```

graph LR
    HTTP2[HTTP/2] --- CO[连接优化]
    HTTP2 --- FE[功能扩展]
    HTTP2 --- B[基于UDP]
    HTTP2 --- PL[编程语言]
    HTTP2 --- WS[WebService]
    HTTP2 --- VAF[VAF]
    
    CO --- OCSP[OCSP]
    CO --- SSL[SSL/TLS]
    
    FE --- SPDY[SPDY]
    FE --- HPACK[HPACK]
    FE --- SP[Server Push]
    FE --- GRPC[gRPC]
    
    B --- QUIC[QUIC]
    
    PL --- CSS[CSS]
    PL --- PHP[PHP]
    PL --- PY[Python]
    
    WS --- RESTful[RESTful]
    WS --- SOAP[SOAP]
    WS --- RPC[RPC]
    
    VAF --- ALP[应用层防护]
    VAF --- AC[访问控制]
    
    HTTP3[HTTP/3] --- QUIC
    HTTP3 --- VAF
  
```

```

graph LR
    subgraph Encoding_Group [编码]
        B64[Base64]
        C[chunked]
    end
    subgraph Compression_Group [压缩]
        G[gzip]
        D[deflate]
    end
    B64 --- C --- Enc[编码]
    G --- D --- Comp[压缩]
    Enc --- Comp --- Enc
    Enc --- WS[WebSocket]
    WS --- WS
  
```

一步了解的，下一讲我会详细讲解这张图。

课下作业

1. 有一种流行的说法：“HTTP 是用于从互联网服务器传输超文本到本地浏览器的协议”，你认为这种说法对吗？对在哪里，又错在哪里？
2. 你能再说出几个“HTTP 不是什么”吗？

给你的朋友。

01 “超文本”这个词经常会引起误解，让人以为 HTTP 只能传输文本文件，个人觉得可能改名叫“超媒体传输协议”更加恰当。

03 我们通常使用浏览器访问的实际上是万维网 (WWW)，它是互联网 (Internet) 的一部分。

但境况几乎很少有人能区分两省。

透视 HTTP 协议

深入理解 HTTP 协议本质与应用

罗剑锋



新版升级：点击「 请朋友读」，20位好友免费读，邀请订阅更有**现金奖励**。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。