**HTTP协议的详解**

**HTTP协议的基本介绍**

1. HTTP协议（HyperText Transfer Protocol 超文本传输协议）是因特网上应用最为广泛的一种网络传输协议，所有的WWW文件都必须遵守这个标准
2. HTTP默认的端口号为80，HTTPS的端口号为443
3. HTTP是一个属于应用层的面向对象的协议，由请求(**Request**)和响应(**Response**)两种模式构成，是一个标准的客户端服务器模型，由于其简捷、快速的方式，适用于分布式超媒体信息系统
4. HTTP是常基于TCP/IP通信协议来传递数据：以下通过一段专业性话语和一段白话文话语来解释一次HTTP请求操作所发生的事情，以让我们能够更好的理解这一特性

**-> 专业：**HTTP协议在开始工作之前，浏览器会先通过TCP/IP协议与服务器建立起一个连接管道，然后HTTP协议会把需要发送的数据按照HTTP协议的格式组织成一个请求报文，而请求报文在通过网线向服务器发送数据之前，又会把数据按照TCP/IP协议进行重新组织（数据在网线中传播的时候必须要使用TCP/IP协议，TCP/IP协议适用于网络硬件设备传输数据），当数据到达服务器接之后，HTTP协议又会重新把数据按照HTTP协议的格式组织成服务器能够识别的数据

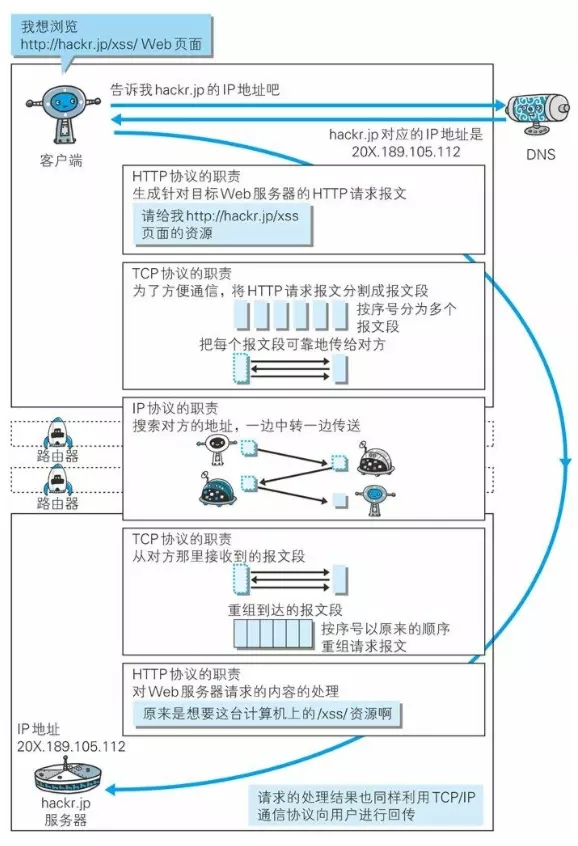
**-> 白话：**生活中的打电话，我们对着电话的麦克风说话（以HTTP协议的格式把数据组织起来），电话接收到我们说的话后，又会把它转换成信号，以信号的方式在电话线中传播至电话的另一端（按照TCP协议的格式重新组织数据，并传递至服务器），电话另一端接收到信号后又会把它转换成我们能听得明白的语言（把数据重新按照HTTP协议的格式组织，便于识别）

**HTTP协议的主要特点**

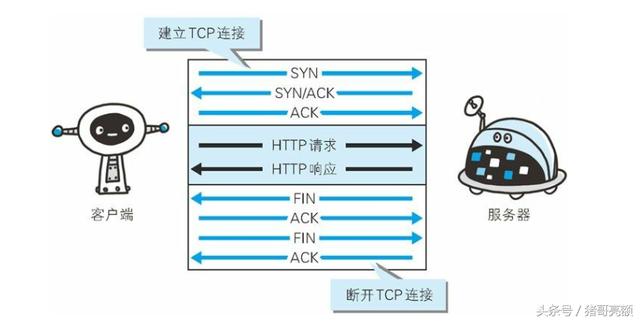
1. 支持B/S及C/S模式
2. 简单快速：客户向服务器请求服务时，只需传送请求方法和路径（请求方法常用的有GET、HEAD、POST，每种方法规定了客户与服务器联系的类型不同）。由于HTTP协议简单，使得HTTP服务器的程序规模小，因而通信速度很快
3. 灵活：HTTP允许传输任意类型的数据对象，正在传输的数据的类型由Content-Type头部字段加以标记
4. 无连接：无连接的含义是限制每次连接只处理一个请求，服务器处理完客户的请求，并收到客户的应答后，即断开连接。采用这种方式可以节省传输时间
5. 无状态：HTTP协议是无状态协议，无状态是指协议对于事务处理没有记忆能力，缺少状态意味着如果后续处理需要前面的信息，则它必须重传，简单来讲我们后续的请求无法获取前面请求的信息，这样可能导致每次连接传送的数据量增大，但在另一方面，服务器不需要先前信息时它的应答就较快。而在实际应用当中并不是完全这样的，我们可以通过引入**Cookie**和**Session**的机制来弥补这一短缺

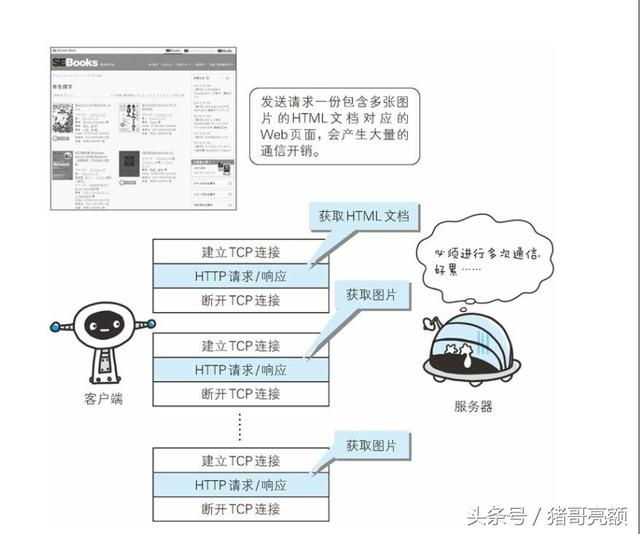
**HTTP协议的工作流程与它的发展与演变**

1. **基本工作流程：**一次HTTP操作称之为一个事物，其基本的工作流程原理如下图

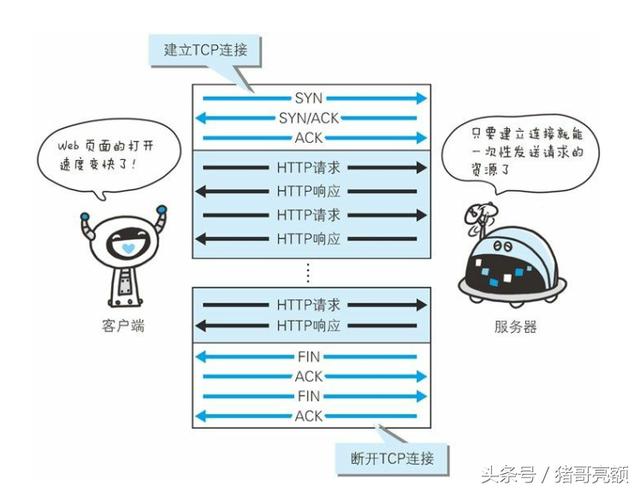


1. **短连接：**HTTP 协议的初始版本中，每进行一次 HTTP 通信就要断开一次 TCP 连接。以当年的通信情况来说，因为都是些容量很小的文本传输，所以即使这样也没有多大问题。可随着 HTTP 的普及，文档中包含大量图片的情况多了起来。比如， 使用浏览器浏览一个包含多张图片的 HTML 页面时，在发送请求访问 HTML 页面资源的同时，也会请求该 HTML 页面里包含的其他资源。因此，每次的请求都会造成无谓的 TCP 连接建立和断开，增加通信量的开销

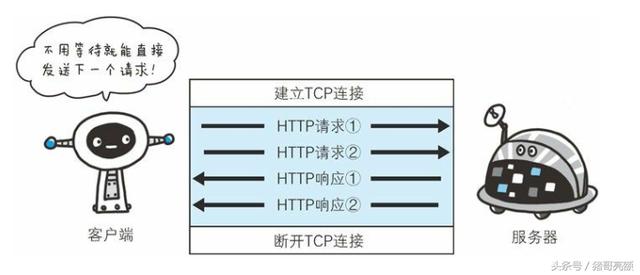




1. **长连接：**为解决短连接上存在的问题，HTTP/1.1 和一部分的 HTTP/1.0 想出了持久连接（HTTP Persistent onnections，也称为 HTTP keep-alive 或 HTTP connection reuse）的方法，它的的特点是，只要任意一端没有明确提出断开连接，则保持 TCP 连接状态。长连接的出现无疑是带来了好处，减少了 TCP 连接的重复建立和断开所造成的额外开销，减轻了服务器端的负载。另外，减少开销的那部分时间，使 HTTP 请求和响应能够更早地结束，这样 Web 页面的显示速度也就相应提高了。在 HTTP/1.1 中，所有的连接默认都是持久连接，但在 HTTP/1.0 内并未标准化。虽然有一部分服务器通过非标准的手段实现了持久连接，但服务器端不一定能够支持持久连接。毫无疑问，除了服务器端，客户端也需要支持持久连接



1. **管线化：**HTTP1.1版本的发布除了长连接，还出现了管线化的技术，长连接使得多数请求以管线化（pipelining）方式发送成为可能。从前发送请求后需等待并收到响应，才能发送下一个请求。管线化技术出现后，不用等待响应亦可直接发送下一个请求。这样就能够做到同时并行发送多个请求，而不需要一个接一个地等待响应了



**HTTP报文**

-> 用于HTTP协议交互的信息被称为 HTTP 报文

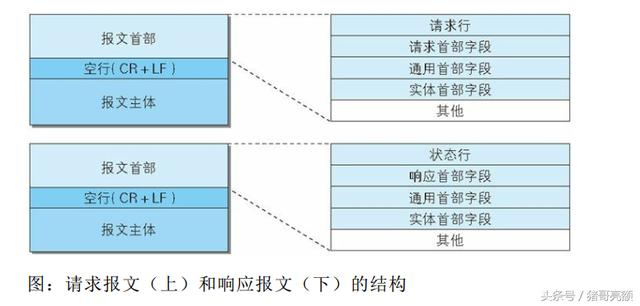
-> 请求端（客户端）的HTTP报文叫做**请求报文**

**->** 响应端（服务器端）的叫做**响应报文**

-> HTTP报文是由多行（用 CR+LF作换行符）数据构成的字符串文本，大致可分为报文首部和报文主体两块。两者由最初出现的空行（CR+LF）来划分，通常，并不一定要有报文主体，如下图所示，这就是HTTP报文内容的结构。



我们也可以更进一步的划分，如下图所示



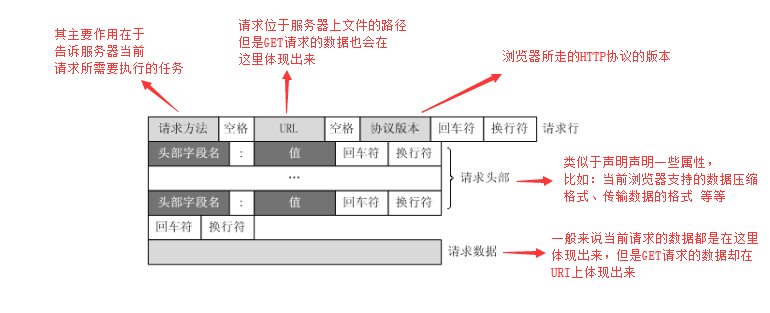
1. **请求报文：客户端发送给服务端的HTTP格式的请求消息**
2. HTTP请求报文由四个部分组成

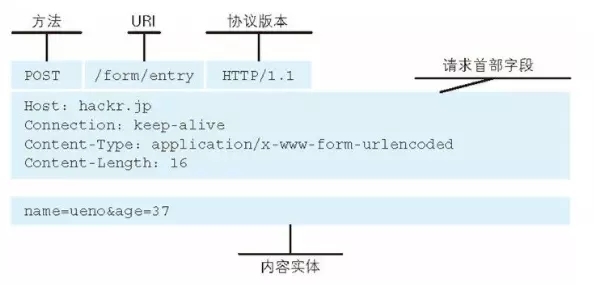
-> 请求行：说明当前的请求方法、URL（**GET请求方法的请求数据附着在这里**）和协议版本

-> 请求头部：类似于声明一些属性，用来说明服务端需要使用的一些附加信息

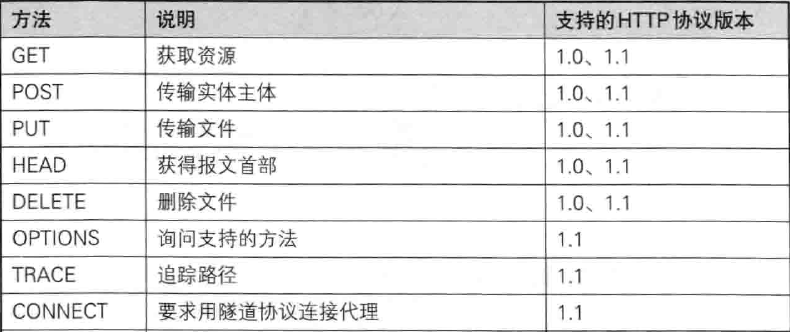
-> **空行：用于分割请求体与请求头**

-> 请求体：所发送的请求数据



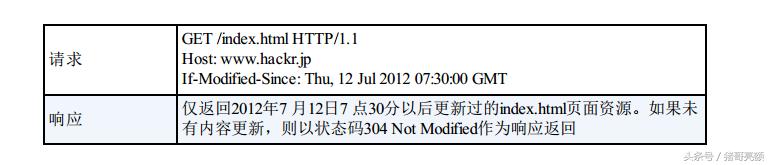


1. 请求方法：位于请求行中的请求方法，是客户端用来告知服务器其动作意图的方法，就像下达命令一样。需要注意的是 **方法名是区分大小写的**，**我们需要用大写字母进行书写**



1. GET（获取资源）：GET方法用来请求访问已被URI识别的资源，也就是指定了服务器处理请求之后响应的内容，该方法提交请求 的数据实体会放在URL的后面，用?来分割URL和传输数据，参数用&连接，**参数以键值对的形式存在**

**案例：浏览器输入地址+回车、搜索、单击超链接（a标签）…**



1. POST（传输实体主体）：POST方法用来传输实体主体，该方法提交的数据实体会放在HTTP请求报文的请求体当中

**案例：注册、上传数据…**



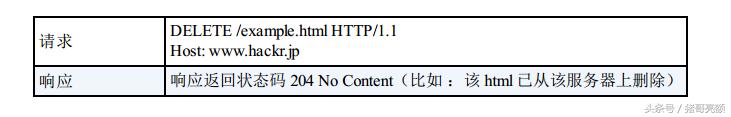
1. PUT（传输文件）：PUT方法用来传输文件，类似FTP协议，文件内容包含在请求报文的实体中，然后请求保存到URL指定的服务 器位置



1. HEAD（获取报文首部）：HEAD方法类似GET方法，但是不同的是HEAD方法不要求返回数据，主要用于确认URI的有效性及资 源更新时间等



1. DELETE（删除文件）：DELETE方法用来删除文件，是与PUT相反的方法。DELETE是要求返回URL指定的资源



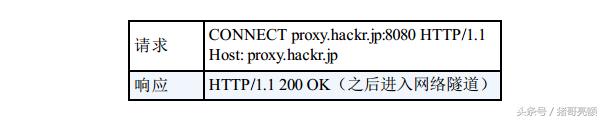
1. OPTIONS（询问支持的方法）：因为并不是所有的服务器都支持规定的方法，为了安全有些服务器可能会禁止掉一些方法例如DELETE、 PUT等，那么OPTIONS就是用来询问服务器支持的方法



1. TRACE（追踪路径）：TRACE方法是让Web服务器将之前的请求通信环回给客户端的方法，这个方法并不常用



1. CONNECT（要求用隧道协议连接代理）：CONNECT方法要求在与代理服务器通信时建立隧道，实现用隧道协议进行TCP通信， 主要使用SSL/TLS协议对通信内容加密后传输



1. POST请求方法和GET请求方法之间的区别
2. GET提交请求的数据实体会放在URL的后面，用?来分割URL和传输数据，参数用&连接，以键值对的形式存在

例如：/index.html?name=wang&login=1

1. POST提交的请求数据则跟在请求报文中的请求体中
2. GET提交的数据长度是有限制的（本身是没有限制的，但是浏览器的URL长度有限制，具体的长度限制视浏览器而定），而POST没有
3. GET提交的数据不安全，因为参数都会暴露在URL上

参考网址：<https://www.cnblogs.com/hyddd/archive/2009/03/31/1426026.html>

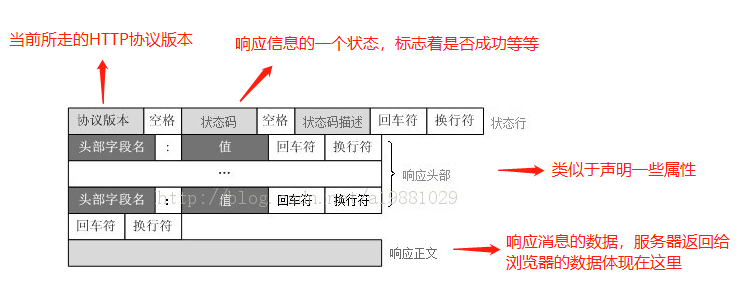
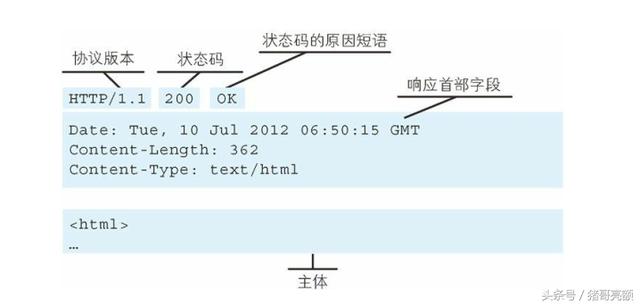
1. **响应报文：服务器接收并处理客户端发过来的请求后所返回的HTTP格式的响应消息**
2. HTTP响应报文由四个部分组成

-> 状态行：说明当前使用的HTTP协议版本、**响应状态码**和简短的状态码解释信息

-> 响应头部：类似于声明一些属性，用来说明浏览器（客户端）需要使用的一些附加信息

-> **空行：用于分割请求体与请求头**

-> 响应体：所发送的响应数据

1. HTTP响应状态码详解

-> 参考网址：<http://tool.oschina.net/commons?type=5>

1. **首部字段：HTTP首部字段是构成HTTP报文最重要的元素之一，在客户端与服务端之前进行信息传递的请求和响应都会使用首部字段，它们会传递一些重要的元信息**
2. **通用首部字段（General Header Fields）：代表请求报文和响应报文都会使用的字段**

****

1. **请求首部字段（Request Header Fields）：是客户端向服务端发送请求时使用的首部字段，包含请求的附加内容、客户端信息、响应内容相关优先级等信息**

****

1. **响应首部字段（Response Header Fields）：是服务端向客户端返回响应时使用的首部字段，包含响应的附加内容，可能也会要求客户端附加额外的内容信息**

****

1. **实体首部字段（Entity Header Fields）：是针对请求报文和响应报文的实体部分使用的首部，包含资源内容更新时间等和实体有关的信息**

**-> 更多关于Content-Type参照：**<Content-Type.doc>

****