HTTP

HTTP协议是Hyper Text Transfer Protocol（超文本传输协议）的缩写,是用于从万维网（WWW:World Wide Web ）服务器传输超文本到本地浏览器的传送协议。

HTTP是一个基于TCP/IP通信协议来传递数据（HTML 文件, 图片文件, 查询结果等）。

HTTP是一个属于应用层的面向对象的协议，由于其简捷、快速的方式，适用于分布式超媒体信息系统。

HTTP协议工作于客户端-服务端架构为上。浏览器作为HTTP客户端通过URL向HTTP服务端即WEB服务器发送所有请求。Web服务器根据接收到的请求后，向客户端发送响应信息。

URL

HTTP使用统一资源标识符（Uniform Resource Identifiers, URI）来传输数据和建立连接。URL是一种特殊类型的URI，包含了用于查找某个资源的足够的信息

URL,全称是UniformResourceLocator, 中文叫统一资源定位符,是互联网上用来标识某一处资源的地址。以下面这个URL为例，

介绍下普通URL的各部分组成：

`http://www.aspxfans.com:8080/news/index.asp?boardID=5&ID=24618&page=1#name`

1. 协议部分：该URL的协议部分为“http：”，这代表网页使用的是HTTP协议。在Internet中可以使用多种协议，如HTTP，FTP等等本例中使用的是HTTP协议。在`HTTP`后面的`//`为分隔符
2. 域名部分：该URL的域名部分为`www.aspxfans.com`。一个URL中，也可以使用IP地址作为域名使用
3. 端口部分：跟在域名后面的是端口，域名和端口之间使用`:`作为分隔符。端口不是一个URL必须的部分，如果省略端口部分，将采用默认端口
4. 虚拟目录部分：从域名后的第一个`/`开始到最后一个`/`为止，是虚拟目录部分。虚拟目录也不是一个URL必须的部分。本例中的虚拟目录是`/news/`
5. 文件名部分：从域名后的最后一个“/”开始到`?`为止，是文件名部分，如果没有`?`,则是从域名后的最后一个`/`开始到`#`为止，是文件部分，如果没有`?`和`#`，那么从域名后的最后一个`/`开始到结束，都是文件名部分。本例中的文件名是`index.asp`。文件名部分也不是一个URL必须的部分，如果省略该部分，则使用默认的文件名
6. 锚部分：从`#`开始到最后，都是锚部分。本例中的锚部分是`name`。锚部分也不是一个URL必须的部分

7. 参数部分：从`?`开始到`#`为止之间的部分为参数部分，又称搜索部分、查询部分。本例中的参数部分为`boardID=5&ID=24618&page=1`。参数可以允许有多个参数，参数与参数之间用“&”作为分隔符。

URI

URI，是uniform resource identifier，统一资源标识符，用来唯一的标识一个资源。

Web上可用的每种资源如HTML文档、图像、视频片段、程序等都是一个来URI来定位的
URI一般由三部组成：

1. 访问资源的命名机制
2. 存放资源的主机名
3. 资源自身的名称，由路径表示，着重强调于资源。

URN

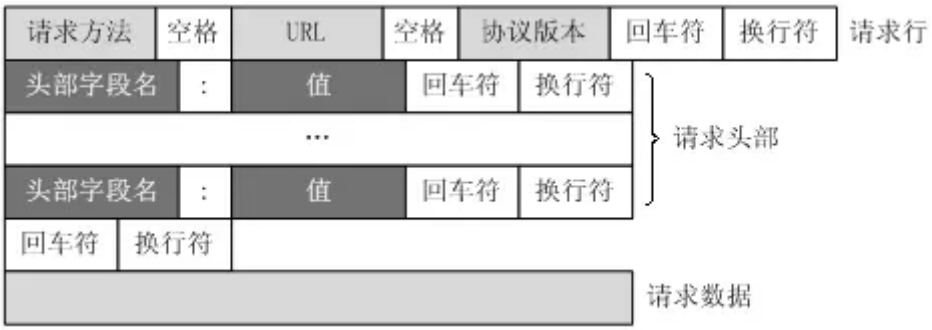
URN，uniform resource name，统一资源命名，是通过名字来标识资源

HTTP 包

Request

客户端发送一个HTTP请求到服务器的请求消息包括以下格式：

请求行（request line）、请求头部（header）、空行和请求数据四个部分组成。



例如：

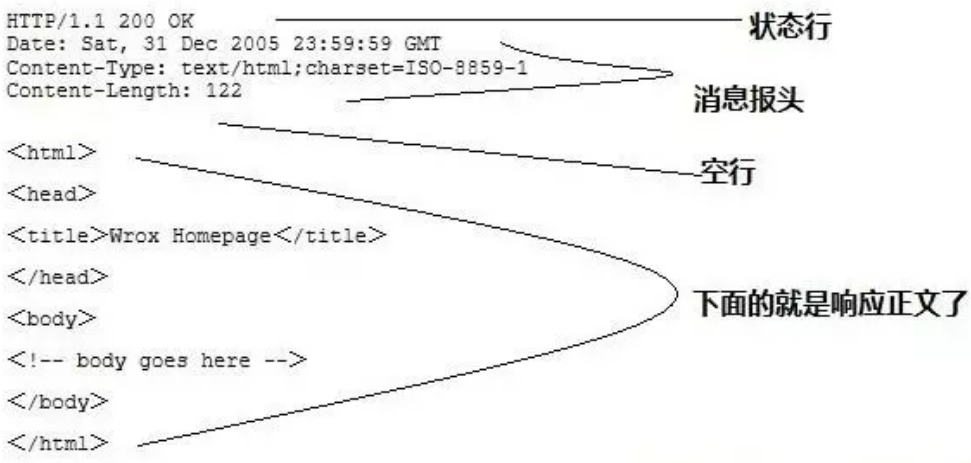
```
POST / HTTP1.1
Host:www.wrox.com
User-Agent:Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR
2.0.50727; .NET CLR 3.0.04506.648; .NET CLR 3.5.21022)
Content-Type:application/x-www-form-urlencoded
Content-Length:40
Connection: Keep-Alive

name=Professional%20Ajax&publisher=Wiley
```

Response

一般情况下，服务器接收并处理客户端发过来的请求后会返回一个HTTP的响应消息。

HTTP响应也由四个部分组成，分别是：状态行、消息报头、空行和响应正文。



例如：

```
HTTP/1.1 200 OK
Date: Fri, 22 May 2009 06:07:21 GMT
Content-Type: text/html; charset=UTF-8

<html>
      <head></head>
      <body>
            <!--body goes here-->
      </body>
</html>
```

状态码

状态代码有三位数字组成，第一个数字定义了响应的类别，共分五种类别:

- 1xx：指示信息--表示请求已接收，继续处理

- 2xx：成功--表示请求已被成功接收、理解、接受
- 3xx：重定向--要完成请求必须进行更进一步的操作
- 4xx：客户端错误--请求有语法错误或请求无法实现
- 5xx：服务器端错误--服务器未能实现合法的请求

常见状态码：

```
200 OK                          //客户端请求成功
400 Bad Request                 //客户端请求有语法错误，不能被服务器所理解
401 Unauthorized                //请求未经授权，这个状态代码必须和WWW-Authenticate报头域
一起使用
403 Forbidden                   //服务器收到请求，但是拒绝提供服务
404 Not Found                   //请求资源不存在，eg：输入了错误的URL
500 Internal Server Error       //服务器发生不可预期的错误
503 Server Unavailable          //服务器当前不能处理客户端的请求，一段时间后可能恢复正常
```

请求方式

根据HTTP标准，HTTP请求可以使用多种请求方法。
HTTP1.0定义了三种请求方法： GET, POST 和 HEAD方法。
HTTP1.1新增了五种请求方法：OPTIONS, PUT, DELETE, TRACE 和 CONNECT 方法。

请求方式	说明
GET	请求指定的页面信息，并返回实体主体。
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	向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST请求可能会导致新的资源的建立和/或已有资源的修改。
PUT	从客户端向服务器传送的数据取代指定的文档的内容。
DELETE	请求服务器删除指定的页面。
CONNECT	HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。
OPTIONS	允许客户端查看服务器的性能。
TRACE	回显服务器收到的请求，主要用于测试或诊断。