# CS 580 Client-Server Programming

## Fall Semester, 2000

### Doc 6 HTTP

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>URI</td>
<td>2</td>
</tr>
<tr>
<td>HTML</td>
<td>3</td>
</tr>
<tr>
<td>HTTP</td>
<td>6</td>
</tr>
<tr>
<td>HTTP Message Format</td>
<td>7</td>
</tr>
<tr>
<td>Client Request</td>
<td>8</td>
</tr>
<tr>
<td>Full-Request</td>
<td>9</td>
</tr>
<tr>
<td>Server Response</td>
<td>11</td>
</tr>
<tr>
<td>Request Methods</td>
<td>17</td>
</tr>
</tbody>
</table>

## References


- Hypertext Transfer Protocol -- HTTP/1.1, Fielding, Gettys, Mogul, Masinter, Leach, Berners-Lee, [rfc2616](http://www.ietf.org/rfc/rfc2616)


## Reading

- HTTP/1.0 [rfc1945](http://www.ietf.org/rfc/rfc1945)

---

**Copyright ©, All rights reserved.**

URI

URI = Uniform Resource Identifiers
URL = Uniform Resource Locator

gopher://gopher.yoyodyne.com/

news:rec.gardening

http://www.yoyodyne.com/pub/foobar.html


Common Internet Scheme Syntax

URL schemes that involve the direct use of an IP-based protocol to a specified host on the Internet use a common syntax for the scheme-specific data:

//<user>:<password>@<host>:<port>/<url-path>
HTML
Some Buzz Words

WWW
World Wide Web (or Web, for short)

SGML
Standard Generalized Markup Language
this is a standard for describing markup languages

DTD
Document Type Definition
this is a specific markup language, written using SGML

HTML
HyperText Markup Language
HTML is a SGML DTD.

HTML uses markup tags to tell the Web browser how to display the text

XML
Extensible Markup Language

XHTML
XML + HTML 4.0
What is HTML?

HTML is a language for describing structured documents

HTML does not describe page layout

Web browsers use HTML to render & display a document

HTML versions

- HTML 1.0: very limited
- HTML 2.0: addition of forms (rfc1866.txt)
- HTML 3.0: failed attempt at standardization
- HTML 3.2: prime candidate for standard. REC-html32:
- HTML 4.0: REC-html40-19980424
  - style sheets
  - scripting
  - frames
  - embedding objects
  - richer tables
  - enhancements to forms

Other developments:

- **CSS** (Cascading Style Sheets)

- **XSL** (Extensible Stylesheet Language)
  Used to transform XML documents
Basic HTML Document

<HTML>
<HEAD>
<TITLE>Sample HTML Document</TITLE>
</HEAD>

<BODY>
This is a document
</BODY>
</HTML>
HTTP

• Stateless (http 1.0)
• Object-oriented protocol

   The typing and negotiation of data representation, allows systems to be built independently of the data being transferred

Assigned port 80

**Basic Server-Client Interaction (http 1.0)**

Client: Open connection

Server: Accept/Reject connection

Client: Send request

Server: Send response to request

Connection closed
HTTP Message Format

HTTP-message = Simple-Request (HTTP/0.9 messages) | Simple-Response |
                          | Full-Request          (HTTP/1.0 messages) |
                          | Full-Response         |

Full-Request = Request-Line *
                  ( General-Header | Request-Header | Entity-Header ) 
                  CRLF 
                  [ Entity-Body ]

Full-Response = Status-Line *
                    ( General-Header | Request-Header | Entity-Header ) 
                    CRLF 
                    [ Entity-Body ]

HTTP-header = field-name ":" [ field-value ] CRLF

Entity-Body = *OCTET
Client Request

Request = Simple-Request | Full-Request
Simple-Request = "GET" SP Request-URI CRLF

Simple-Request Example

rohan 11-> telnet www.eli.sdsu.edu 80
Trying 130.191.226.80...
Connected to www.eli.sdsu.edu.
Escape character is '^]'.
GET /courses/fall00/cs580/index.html
<HTML>
<HEAD>
    <TITLE>CS 580: Course Web Site</TITLE>
</HEAD>
<BODY BGCOLOR="#FFFFFF">

<TABLE BORDER=0 WIDTH="100%">

…stuff removed…

</sub>Visitors since 21-Aug-00
</center>
</BODY>
</HTML>
Connection closed by foreign host.
Full-Request

Full-Request = Request-Line
   *( General-Header | Request-Header | Entity-Header )
CRLF
[ Entity-Body ]

Request-Line = Method SP URI SP HTTP-Version CRLF

rohan 13-> telnet www.eli.sdsu.edu 80
Trying 130.191.226.80...
Connected to www.eli.sdsu.edu.
Escape character is "^]".
GET /courses/fall00/cs580/index.html HTTP/1.0

HTTP/1.1 200 OK
Date: Tue, 05 Sep 2000 19:31:14 GMT
Server: Apache/1.3.9 (Unix) PHP/3.0.12
Last-Modified: Mon, 04 Sep 2000 21:03:56 GMT
ETag: "14c199-7e8-39b40e3c"
Accept-Ranges: bytes
Content-Length: 2024
Connection: close
Content-Type: text/html
X-Pad: avoid browser bug

<HTML>
<HEAD>
   <TITLE>CS 580: Course Web Site</TITLE>
</HEAD>
… stuff removed here…
Connection closed by foreign host.

Note 2 CRLF are needed to end the full request
HTTP 1.1 Example

rohan 14-> telnet www.eli.sdsu.edu 80
Trying 130.191.226.80...
Connected to www.eli.sdsu.edu.
Escape character is '^[].
GET /courses/fall00/cs580/index.html HTTP/1.1
Connection: close
Host: www.eli.sdsu.edu

HTTP/1.1 200 OK
Date: Tue, 05 Sep 2000 22:41:26 GMT
Server: Apache/1.3.9 (Unix) PHP/3.0.12
Last-Modified: Mon, 04 Sep 2000 21:03:56 GMT
ETag: "14c199-7e8-39b40e3c"
Accept-Ranges: bytes
Content-Length: 2024
Connection: close
Content-Type: text/html
X-Pad: avoid browser bug

<HTML>
<HEAD>
    <TITLE>CS 580: Course Web Site</TITLE>
</HEAD>
…stuff removed here…
</BODY>
</HTML>
Connection closed by foreign host.
Server Response

Example Full-response

HTTP/1.0 200 Document follows
MIME-Version: 1.0
Server: CERN/3.0
Date: Thursday, 21-Mar-96 17:00:45 GMT
Content-Type: text/html
Content-Length: 2686
Last-Modified: Tuesday, 27-Feb-96 05:34:12 GMT

field-name    field-value
MIME-Version: 1.0
Server: CERN/3.0
Date: Thursday, 21-Mar-96 17:00:45 GMT
Content-Type: text/html
Content-Length: 2686
Last-Modified: Tuesday, 27-Feb-96 05:34:12 GMT
What is the big Deal?

Name-Value Pairs

What are the data fields in this?

1.0; CERN/3.0; Thursday, 21-Mar-96 17:00:45 GMT; text/html; 2686; Tuesday, 27-Feb-96 05:34:12 GMT

What are the data fields in this?

MIME-Version: 1.0
Server: CERN/3.0
Date: Thursday, 21-Mar-96 17:00:45 GMT
Content-Type: text/html
Content-Length: 2686
Last-Modified: Tuesday, 27-Feb-96 05:34:12 GMT

Which is Safer?

Which is Easier to Parse?
Name -Value Pairs are Good

Does Order Matter?

MIME-Version: 1.0
Server: CERN/3.0
Date: Thursday, 21-Mar-96 17:00:45 GMT
Content-Type: text/html
Content-Length: 2686
Last-Modified: Tuesday, 27-Feb-96 05:34:12 GMT

Server: CERN/3.0
Content-Type: text/html
MIME-Version: 1.0
Content-Length: 2686
Last-Modified: Tuesday, 27-Feb-96 05:34:12 GMT
Date: Thursday, 21-Mar-96 17:00:45 GMT

Extending Protocols

MIME-Version: 1.0
Server: CERN/3.0
Date: Thursday, 21-Mar-96 17:00:45 GMT
Content-Type: text/html
Forwarded: by http://rohan.sdsu.edu/ for cs.sdsu.edu
Content-Length: 2686
WhitneyInfo: Hi Mom
Last-Modified: Tuesday, 27-Feb-96 05:34:12 GMT
Name -Value Pairs are Everywhere

Data Files
Which is easier for a program to parse?
Which is safer

<table>
<thead>
<tr>
<th>name</th>
<th>course</th>
<th>hwork</th>
<th>exam1</th>
<th>exam2</th>
<th>final</th>
<th>as1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allen, Sally</td>
<td>87</td>
<td>92</td>
<td>85</td>
<td>55</td>
<td>74</td>
<td>10</td>
</tr>
<tr>
<td>Battista, Joe</td>
<td>92</td>
<td>98</td>
<td>98</td>
<td>55</td>
<td>78</td>
<td>10</td>
</tr>
<tr>
<td>Biag, Sam</td>
<td>83</td>
<td>91</td>
<td>78</td>
<td>51</td>
<td>72</td>
<td>8</td>
</tr>
<tr>
<td>Chen, Pete</td>
<td>89</td>
<td>92</td>
<td>89</td>
<td>57</td>
<td>79</td>
<td>10</td>
</tr>
<tr>
<td>Chen, Roger</td>
<td>74</td>
<td>68</td>
<td>59</td>
<td>61</td>
<td>55</td>
<td>10</td>
</tr>
</tbody>
</table>

lastName:Allen, lastName:Sally, course:87, hwork:92, exam1:85, exam2:55, final:74, as1:10
lastName:Battista, lastName:Joe, course:92, hwork:98, exam1:98, exam2:55, final:78, as1:10
lastName:Baig, lastName:Sam, course:83, hwork:91, exam1:78, exam2:51, final:72, as1:8
lastName:Chen, lastName:Pete, course:89, hwork:92, exam1:89, exam2:57, final:79, as1:10
lastName:Chen, lastName:Roger, course:74, hwork:68, exam1:59, exam2:61, final:55, as1:10
Name -Value Pairs are Your Friends

Don't program without them
Big Issue: In-line data

If we send binary data or data of unknown format how does receiver know when the data ends?

**POP solution**

Use termination sequence

Insure that termination sequence does not occur in data

**HTTP Solution**

Full-Response = Status-Line  
*General-Header  
*Response-Header  
*Entity-Header  
CRLF  
[ Entity-Body ]

Send length of data to be sent in header
Request Methods

Method = "GET" | "HEAD" | "PUT" | "POST"  
| "DELETE" | "LINK" | "UNLINK"  
| extension-method

All HTTP/1.0 servers must support GET and HEAD

Servers should return the Status-Code

"501 Not Implemented"

if the method is unknown.

GET

Retrieves whatever item is identified by the URI.

The URI can refer to a data-producing process, or a script

The produced data which shall be returned as the Entity-Body

HEAD

Identical to GET except that the server must not return any Entity-Body in the response
POST

Request that the origin server accept the item enclosed in the request as a new subordinate of the resource identified by the URI

Allows a uniform function to:

Annotation of existing documents;

Posting a message to a bulletin board topic, newsgroup, mailing list, or similar group of articles;

Providing a block of data (usually a form) to a data-handling process, or a script, which can be run by such a process;

Extending a document during authorship
These are not always supported

Why?

**PUT**

The enclosed item in the request is to be stored under the supplied URI

**DELETE**

Requests that the server delete the resource identified by the given URI

**LINK**

Establishes one or more Link relationships between the existing resource identified by the URI and other existing resources

**UNLINK**

UNLINK method removes one or more Link relationships from the existing resource identified by the URI