CS 580 Client-Server Programming
Spring Semester, 2005
Doc 14 HTTP & POP
Contents

References

Hypertext Transfer Protocol - HTTP/1.0, Berners-Lee, Fielding, Nielson, rfc1945, http://www.w3.org/Protocols/rfc1945/rfc1945

Hypertext Transfer Protocol -- HTTP/1.1, Fielding, Gettys, Mogul, Masinter, Leach, Berners-Lee, rfc2616, http://www.w3.org/Protocols/rfc2616/rfc2616.html

Uniform Resource Identifiers (URI): Generic Syntax, Berners-Lee, Fielding, Masinter, rfc2396 http://www.ietf.org/rfc/rfc2396.txt

RFC 1939, http://www.ietf.org/rfc/rfc1939.txt

Reading

HTTP/1.0 rfc1945, http://ftp.ics.uci.edu/pub/ietf/http/rfc1945.html

Post Office Protocol RFC 1939, http://www.ietf.org/rfc/rfc1939.txt

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>
</HTML>
… 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.
```

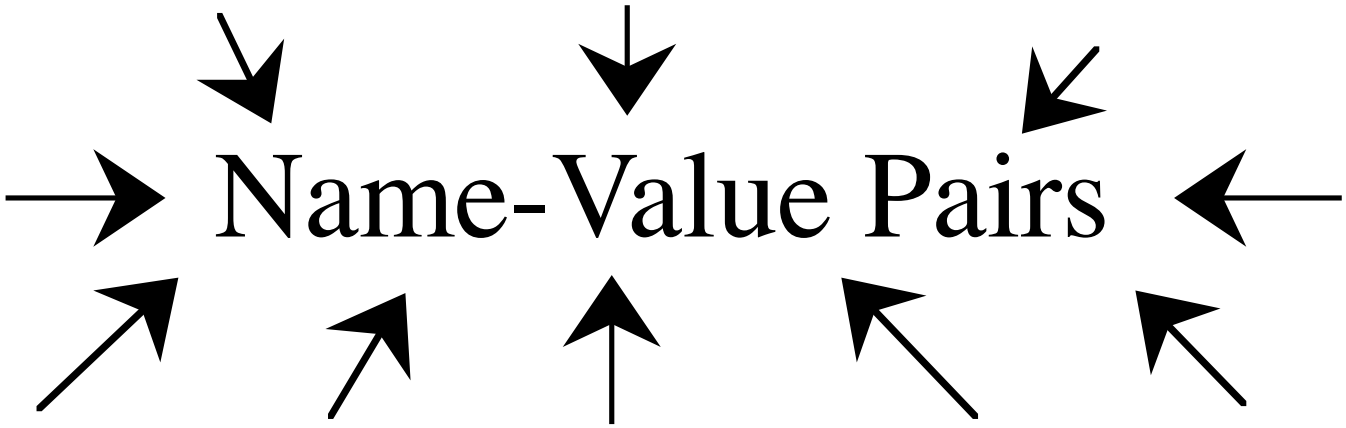
Server Response

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?



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

Cookies were added to HTTP by adding a new name-value pair

Clients/servers that were not programmed for cookies ignored the new name-value pair

Name -Value Pairs are Everywhere

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

Allen, Sally	87	92	85	55	74	10
Battista, Joe	92	98	98	55	78	10
Biag, Sam	83	91	78	51	72	8
Chen, Pete	89	92	89	57	79	10
Chen, Roger	74	68	59	61	55	10

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

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 and Parameters

Most languages use positional matching for parameters

“The cat in the hat came back”.substring(2, 6);

Smallktalk uses name-value pairs

‘The cat in the hat came back’ copyFrom: 2 to: 6

copyFrom:to: is on method

Keyword (name)	Parameter (value)
copyFrom:	2
to:	6

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

POP3 Protocol

Purpose: Allow PC's, Macs, etc. to download mail from server

Port number 110

Protocol uses ASCII only

Command Format

Format of commands to server

keyword blank $argument_1$ [blank $argument_k$] CRLF

| keyword | = 3, 4 characters

| argument | <= 40 characters

keyword and arguments are separated by single space character

Server Response

Status keyword additionalInfo

Status is either "+OK" or "-ERR0.3."

A single line response ends in CRLF

If response requires more than one line:

- Each line ends in a CRLF
- The response ends in CRLF.CRLF
- If a line starts with a "." prepend a "." to it

Timeouts

A POP3 server may have an autologout timer

A server must wait at least 10 minutes before timing out a client

The POP3 server on cs.sdsu.edu times out in 2 minutes

States

AUTHORIZATION

Must log in with password before entering transaction state

TRANSACTION

Client can request actions of server, get mail for example

UPDATE

Updates mail box to reflect actions taken in transaction state

Commands
AUTHORIZATION

Server acknowledges connection from client with

+OK "message"

+OK UCB Pop server (version 2.1.2-R3) at sciences.sdsu.edu starting.

Commands: USER, PASS, APOP, QUIT

USER PASS

Combination is used to progress to transaction state

USER must come first
PASS or QUIT must come after USER

Example

Ti 38->**telnet cs.sdsu.edu 110**
Trying 130.191.226.116...
Connected to cs.sdsu.edu.
Escape character is '^]'.
+OK QPOP (version 3.1.2) at sciences.sdsu.edu starting.
USER whitney
+OK Password required for whitney.
PASS typeYourPasswordHere
+OK whitney has 116 visible messages (0 hidden) in 640516 octets.

TRANSACTION

Commands: STAT, LIST, RETR, RSET, QUIT

STAT

Arguments: none

Returns "+OK" numberOfMessages SizeOfMail

Example

STAT
+OK 22 45595

LIST

Arguments: a message-number (optional)

Returns: size of message in octets

Examples:

LIST 2
+OK 2 3064

LIST
+OK 116 visible messages (640516 octets)
1 2980
2 3064 (message 3 - 116 deleted to save space)
116 1290
.

RETR

Arguments: a message-number

Returns: the message

Example:

RETR 21

+OK 825 octets
Received: from [130.191.9.18] (ebb2p9.sdsu.edu [130.191.9.18]) by sciences.sdsu.edu (4.1/8.6.10) with SMTP id UAA29486 for <whitney@saturn.sdsu.edu>; Mon, 11 Mar 1996 20:16:07 -0800 (PST)
X-Sender: whitney@cs.sdsu.edu (Unverified)
Message-Id: <v02110100ad6aaaf097b6@[130.191.9.70]>
Mime-Version: 1.0
Content-Type: text/plain; charset="us-ascii"
Date: Mon, 11 Mar 1996 20:16:50 -0800
To: whitney@saturn.sdsu.edu
From: whitney@saturn.sdsu.edu (Roger Whitney)
Subject: Sample Mail
X-UIDL: 826604201.000

this is a test
..
the end

Roger Whitney Math & Computer Science Dept.
whitney@cs.sdsu.edu San Diego State University
http://www.eli.sdsu.edu San Diego, CA 92182-7720
(619) 594-3535
(619) 594-6746 (fax)

.

DELE

Arguments: a message-number to delete

Returns: a confirmation of deletion

Marks a message to be deleted

NOOP

Arguments: none

Returns: a positive response

Does nothing

QUIT

Arguments: none

Returns: a positive response

Send POP3 server to UPDATE state

UPDATE State

Updates mail box to reflect transactions taken during the transaction state, then logs user out

If session ends by any method except the QUIT command during the transaction state, the update state is not entered

Optional POP3 Commands
TOP

Arguments: a message-number and number of lines to return

Returns: Requested lines of indicated message

State allowed in: transaction

UIDL

Arguments: a message-number (optional)

Returns: a unique-id listing for message

State allowed in: transaction

Examples:
UIDL 1
+OK 1 826312760.001

UIDL
+OK uidl command accepted.
1 826312760.001
2 826312760.006
3 826493796.004
etc.
.

APOP

Arguments: a mailbox and a MD5 digest string

State allowed in: authorization

Action: If MD5 string is correct move to transaction state