Hypertext Transfer Protocol HTTP/1.x

Roy T. Fielding

<fielding@ics.uci.edu>
University of California, Irvine

34th Internet Engineering Taskforce
HTTP Working Group

What We Are Doing

Defining standards for HTTP/1.0, 1.1, and 1.2

- **HTTP/1.0 <draft-ietf-http-v10-spec-04>**
  - **interoperability requirements for the commonly and consistently implemented features in current practice**
  - **move to proposed standard now?**
- **HTTP/1.1 <draft-ietf-http-v11-spec-00>**
  - **improvements for persistent connections, hierarchical caching, content negotiation, partial retrieval, request preconditions, digest and proxy authentication**
  - **much work left to do, proposed standard by March IETF**
- **HTTP/1.2**
  - **whatever is left that still fits within the MIME-like message model**
  - **progress as small, independent drafts**

HTTP 1.x Design Constraints

Speed

Simplicity

Extensibility

Data Independence

Access Authentication

Hierarchical Caching

Content Negotiation

Speed

Emphasis is on immediacy of traversal response across a global network

- **use response caches**
- **send important info first**
- **reduce round-trips**
- ***but without any transport hacks***

Simplicity

- **ASCII control structure**
  - Readability via TELNET is a feature!
- **Mail entity reuse**
  - header containing metainformation
  - body containing data
- **Request/response sequence**
  - no asynchronous or interleaved responses
  - pipelined requests are okay
- **Few requirements for minimal compliance**
  - it should still be possible to implement a minimal server in a single day's work

Extensibility

- **Methods**
  - each method defines an operation to be applied by a resource
- **Uniform Resource Identifiers (URI)**
  - resources are identified via the Request-URI
- **Header Fields**
  - general header fields give information about the message
  - request header fields assign parameters to the request
  - response header fields characterize the response
  - entity header fields provide metainformation
- **Response Status Codes**
  - 3 digits indicate status independent of language
  - First digit defines the class of response

Data Independence

- **Transfer Codings**
  - define encodings applied to the message entity for transfer
- **Content Codings**
  - define encodings applied by the resource over media type
- **Media Types**
  - define the format/purpose of the enclosed data entity
- **Character Sets**
  - define the character encoding for textual media types
- **Language Tags**
  - define the natural language(s) for which the entity is intended

Access Authentication

- **Access controlled by the server**
  - for each resource (URI) or resource hierarchy
  - by each method
- **Basic**
  - sends user:password in clear (with only base64 encoding)
  - should be phased-out as soon as possible
- **Digest**
  - sends digested hash of user:password + anti-replay info
  - not truly secure, but much better than Basic, and about the best that can be accomplished without a patent license and without encountering export controls
  - move to experimental now, proposed standard in HTTP/1.1

Hierarchical Caching

- **Must design for intermediaries**
  - we cannot assume user agent can access origin server
  - proxies, gateways, and tunnels are all possible
- **Data distributed by lazy flood**
  - responses delivered to those who ask
  - cachable responses stored by all caches on response chain
- **Minimize stateful requests/responses**
  - statefulness prevents sharing of responses
  - stateful data should be segregated from stateless data whenever possible
- **Identify when unsafe to cache**
  - make it easy for good providers to do good
  - identify stateful data in headers and/or body

Content Negotiation

Resources are independent of format

- **Preemptive Negotiation**
  - client sends what is acceptable/preferred
  - server chooses one
- **Reactive Negotiation**
  - server responds with multiple choices
  - client chooses one

Major Changes in HTTP/1.1

Request-URI

- **allow asterisk (“*”) to represent entire server**

New Methods

- **OPTIONS: tell me about ...**
- **PUT : as before, but now two-phased**
- **PATCH: like PUT , but contains differences**
- **COPY , MOVE, and DELETE**
- **LINK and UNLINK**
- **TRACE: respond with a reflection of request**
- **WRAPPED: encapsulate other request(s)**

Major Changes in HTTP/1.1

New Status Codes

100 Continue	405 Method Not Allowed
101 Switching Protocols	406 None Acceptable
205 Reset Content	407 Proxy Authentication Required
206 Partial Content	408 Request Timeout
300 Multiple Choices	409 Conflict
303 See Other	410 Gone
305 Use Proxy	411 Length Required
	412 Unless True [Condition Failed]
	504 Gateway Timeout

Major Changes in HTTP/1.1

New General Header Fields

Cache-Control: identify what is and is not cachable in any message

Connection, Keep-Alive: support for persistent connections

Forwarded: make proxies and gateways visible

Upgrade: allow the client and server to switch application protocol(s)

New Request Header Fields

Accept, Accept-Charset, Accept-Encoding, Accept-Language

Host: required on every request to identify original host from URL

Proxy-Authorization: provide authentication credentials to a proxy

Range: allow requests for only a partial range of the normal response

Unless (or IF): places a precondition on any request

Major Changes in HTTP/1.1

New Response Header Fields

Proxy-Authenticate: allows a proxy to challenge authentication

Public: allows the server to declare what methods are implemented

Retry-After: used with 503 (service unavailable) response

New Entity Header Fields

Link, Title, URI, Content-Language, Content-MD5

Content-Range: implies that the entity is only a part of the whole resource

Content-Version, Derived-From: support for opaque versioning

Content-Location: defines the actual URL of the entity

Content-Base: defines the base URL for parsing relative URLs (RFC 1808)

Transfer-Encoding: allows the transfer agent(s) to add encodings

Logic Bags for Request Preconditions

A General Mechanism for Logical Expressions

```
logic-bag    = "{" expression "}"
expression   = ( log-op 1*logic-bag )
             |  ( rel-op 1*field-tuple )
             |  ( "def" 1*field-name )
log-op       = "and" | "or" | "xor" | "not"
rel-op       = "eq" | "ne" | "lt" | "le" | "ge" | "gt" | "in"
field-tuple  = "{" field-name ( bag | token | quoted-string ) "}"
```

```
{or {ne {Content-MD5
          "Q2hlY2sgSW50ZWdyaXR5IQ=="}}
    {ne {Content-Length 10036}}
    {ne {Content-Version "12.4.8"}}
    {gt {Last-Modified
          "Mon, 04 Dec 1995 01:23:45 GMT"}}}
```