Các trường Header trong HTTP

Các trường Header cung cấp thông tin được yêu cầu về yêu cầu hoặc phản hồi, hoặc về đối tượng được gửi trong phần thân thông báo. Có 4 kiểu của Header thông báo HTTP:

- **Kiểu chung (General-Header):** Các trường Header này có khả năng ứng dụng chung cho cả các thông báo yêu cầu và phản hồi.
- **Kiểu yêu cầu (Request-Header):** Các trường Header này có khả năng ứng dụng chỉ cho các thông báo yêu cầu.
- **Kiểu phản hồi (Response-Header):** Các trường Header này chỉ có khả năng áp dụng cho các thông báo phản hồi.
- **Kiểu thực thể (Entity-Header):** Các trường này xác định thông tin về thân-thực thể hoặc, nếu không có phần thân nào hiển thị, về nguồn được nhận diện bởi yêu cầu.

General Header

Trường Cache-Control

Trường Header chung **Cache-Control** được sử dụng để xác định các chỉ dẫn mà **PHẢI** được tuân theo bởi tất cả các hệ thống bộ nhớ ẩn. Cú pháp như sau:

```
Cache-Control : cache-request-directive|cache-response-directive
```

Một Client hoặc Server có thể sử dụng Header chung **Cache-Control** để xác định các tham số cho bộ nhớ ẩn hoặc yêu cầu các loại cụ thể của tài liệu từ bộ nhớ ẩn. Các chỉ dẫn bộ nhớ ẩn được xác định trong một danh sách được phân biệt bởi dấu phảy. Ví dụ:

```
Cache-control: no-cache
```

Bảng dưới liệt kê các chỉ dẫn yêu cầu bộ nhớ ẩn quan trọng mà có thể được sử dụng bởi**Client** trong yêu cầu HTTP của nó:

STT	Chỉ dẫn yêu cầu bộ nhớ ẩn và miêu tả
1	**no-cache** Một bộ nhớ ẩn phải không sử dụng phản hồi để làm thỏa mãn một yêu cầu theo sau mà

	không tái xác nhận thành công với Server ban đầu.
2	**no-store** Bộ nhớ ẩn không nên lưu giữ bất cứ thứ gì về yêu cầu Client hoặc phản hồ Server.
3	**max-age = giây (s)** Chỉ ra rằng Client đang muốn chấp nhận một phản hồi mà thời gian của nó không lớn hơn thời gian đã xác định bằng giây (s).
4	**max-stale [tính bằng giây]** Chỉ ra rằng Client đang muốn chấp nhận một phản hồi mà đã vượt thời gian mãn hạn. Nếu số giây được cung cấp, nó phải không là hết hạn bởi nhiều hơn thời gian đó.
5	**min-fresh = giây** Chỉ ra rằng Client đang muốn chấp nhận một phản hồi mà thời gian sống khỏe của nó là không ít hơn tuổi hiện tại của nó cộng với thời gian đã xác định bằng giây.
6	**no-transform** Không chuyển đổi phần thân đối tượng.
7	**only-if-cached** Không lấy dữ liệu mới. Bộ nhớ ẩn có thể gửi một tài liệu chỉ khi nó ở trong bộ nhớ ẩn, và không nên liên hệ với Server ban đầu để xem xét nếu một bản sao mới hơn tồn tại.

Các chỉ dẫn phản hồi bộ nhớ ẩn quan trọng sau đây có thể được sử dụng bởi **Server** trong phản hồi của nó:

STT	Chỉ dẫn phản hồi bộ nhớ ẩn và Miêu tả
1	**public** Chỉ ra rằng phản hồi có thể được giữ trong bộ nhớ ẩn bởi bất cứ bộ nhớ ẩn nào.
2	**private**

	Chỉ ra rằng tất cả hoặc một phần của thông báo phản hồi được xem như là cho một người sử dụng đơn và phải không được giữ trong bộ nhớ ẩn bởi một bộ nhớ ẩn được chia sẻ.
3	**no-cache** Một bộ nhớ ẩn phải không sử dụng phản hồi để thỏa mãn một yêu cầu theo sau mà không tái xác nhận thành công với Server ban đầu.
4	**no-store** Bộ nhớ ẩn không nên lưu bất cứ gì về yêu cầu Client hoặc phản hồi Server.
5	**no-transform** Không chuyển đổi phần thân đối tượng.
6	**must-revalidate** Bộ nhớ ẩn phải xác minh trạng thái của các tài liệu đã cũ trước khi sử dụng nó và các tài liệu đã mãn hạn không nên được sử dụng.
7	**proxy-revalidate** Chỉ dẫn tái xác nhận ủy quyền có cùng ý nghĩa với chỉ dẫn must-revalidate, ngoại trừ nó không áp dụng tới các bộ nhớ ẩn user agent không được chia sẻ.
8	**max-age = giây** Chỉ ra rằng Client đang muốn chấp nhận một yêu cầu mà tuổi của nó không lớn hơn thời gian đã xác định bằng giây.
9	**s-maxage = giây** Tuổi tối đa được xác định bởi chỉ dẫn này vượt quá tuổi tối đa đã xác định bởi hoặc chỉ dẫn max-age hoặc Expires Header. Chỉ dẫn s-maxage luôn luôn được bỏ qua bởi một bộ nhớ cá nhân.

Trường Connection

Trường Header chung **Connection** cho phép người gửi xác định các chức năng mà được mong ước cho kết nối cụ thể đó và phải không được giao tiếp bởi các trạm ủy nhiệm qua các kết nối xa hơn. Dưới đây là cú pháp đơn giản cho sử dụng Connection Header:

```
Connection : "Connection"
```

HTTP/1.1 xác định rõ chức năng kết nối "**close**" cho người gửi tới tín hiệu mà kết nối sẽ được đóng sau khi hoàn thành phản hồi. Ví dụ:

```
Connection: close
```

Theo mặc định, HTTP 1.1 sử dụng các kết nối liên tục, nơi mà kết nối không tự động đóng sau khi hoàn thành một giao dịch. Trong khi đó, HTTP 1.0 không có các kết nối liên tục theo mặc định. Nếu một Client 1.0 mong muốn sử dụng các kết nối liên tục, nó sử dụng các tham số **keep-alvie** như sau:

```
Connection: keep-alive
```

Trường Date

Tất cả các nhãn Ngày/Thời gian **PHẢI** được biểu diễn trong GMT, không có trường hợp ngoại trừ. Các ứng dụng HTTP được cho phép được sử dụng bất kỳ 3 sự biểu diễn nhãn Ngày/Thời gian nào sau đây:

```
Sun, 06 Nov 1994 08:49:37 GMT  ; RFC 822, updated by RFC 1123 Sunday, 06-Nov-94
08:49:37 GMT ; RFC 850, obsoleted by RFC 1036 Sun Nov  6 08:49:37 1994       ;
ANSI C's asctime() format
```

Ở đây, định dạng đầu tiên là được sử dụng nhiều nhất.

Trường Pragma

Trường Pragma được sử dụng để bao gồm các chỉ dẫn cụ thể để thực hiện mà có thể áp dụng tới bất kỳ người nhận nào trong chuỗi Yêu cầu/Phản hồi. Ví dụ:

```
Pragma: no-cache
```

Chỉ dẫn chỉ xác định rõ trong HTTP/1.0 là chỉ dẫn không bộ nhớ ẩn và được duy trì trong HTTP/1.1 cho tính tương thích ngược về sau. Không có các chỉ dẫn Pragma mới sẽ được định nghĩa trong tương lai.

Trường Trailer

Giá trị **trường Trailer** chỉ ra rằng thiết lập đã cho của các trường Header biểu diễn trong trailer của một thông báo được mã hóa với mã hóa truyền tải được đóng khối. Dưới đây là cú pháp của trường Trailer:

```
Trailer : field-name
```

Các trường Header thông báo được liệt kê trong trường Trailer phải không bao gồm các trường Header sau:

- Transfer-Encoding
- Content-Length
- Trailer

Trường Transfer-Encoding (Mã hóa truyền tải)

Trường *Transfer-Encoding* này chỉ ra kiểu truyền tải nào được áp dụng tới phần thân thông báo để cho việc truyền tải một cách an toàn giữa người gửi và người nhận. Điều này không giống như **Content-encoding** bởi vì các mã hóa truyền tải là một thuộc tính của thông báo, không phải là của phần thân thông báo. Cú pháp của trường Transfer-Encoding là như sau:

```
Transfer-Encoding: chunked
```

Tất cả các giá trị Transfer-Encoding là không nhạy cảm (không phân biệt hoa-thường).

Trường Upgrade

Trường *Upgrade* này cho phép Client xác định những giao thức giao tiếp thêm vào mà nó hỗ trợ và sẽ được sử dụng nếu Server tìm thấy rằng nó thích hợp để chuyển đổi giao thức. Ví dụ:

```
Upgrade: HTTP/2.0, SHTTP/1.3, IRC/6.9, RTA/x11
```

Trường Upgrade được chờ đợi để cung cấp một kỹ thuật đơn giản cho truyền tải từ HTTP/1.1 tới một số giao thức không tương hợp.

Trường Via

Trường *Via* phải được sử dụng bởi các gateway và các trạm ủy nhiệm để chỉ ra các giao thức trung gian và người nhận. Ví dụ, một thông báo yêu cầu có thể được gửi từ một HTTP/1.0 User agent tới một trạm ủy nhiệm nội bộ được đặt tên mã “fred”, mà sử dụng HTTP/1.1 để chuyển tiếp yêu cầu tới một trạm ủy nhiệm công cộng tại nowhere.com, mà hoàn thành yêu cầu bởi việc chuyển tiếp nó tới Server ban đầu tại www.ics.uci.edu. Yêu cầu được nhận bởi www.ics.uci.edu sẽ có trường Via như sau:

```
Via: 1.0 fred, 1.1 nowhere.com (Apache/1.1)
```

Trường Warning (Cảnh báo)

Trường *Warning* được sử dụng để mang thông tin thêm về trạng thái hoặc sự truyền tải của một thông báo mà có thể không được phản ánh trong thông báo đó. Một sự phản hồi có thể mang nhiều hơn một trường Warning.

```
Warning : warn-code SP warn-agent SP warn-text SP warn-date
```

Các trường Header yêu cầu trên Client

Trường Accept (Chấp nhận)

Trường *Accept* này có thể được sử dụng để xác định các kiểu phương tiện cụ thể mà là có thể chấp nhận cho sự phản hồi. Cú pháp chung là như sau:

```
Accept: type/subtype [q=qvalue]
```

Các kiểu phương tiện có thể được liệt kê phân biệt nhau bởi các dấu phảy và giá trị q tùy ý biểu diễn một mức độ chất lượng có thể chấp nhận để chấp nhận các kiểu trên một phạm vi từ 0 tới 1. Dưới đây là ví dụ:

```
Accept: text/plain; q=0.5, text/html, text/x-dvi; q=0.8, text/x-c
```

Đoạn này có thể được biên dịch như **text/html** và **text/x-c** và là các kiểu phương tiện được ưa thích hơn nhưng nếu chúng không tồn tại, thì sau đó gửi đối tượng **text/x-dvi** , và nếu nó không tồn tại, gửi đối tượng **text/plain**.

Trường Accept-Charset

Trường này có thể được sử dụng để chỉ các bộ thiết lập ký tự nào được chấp nhận cho sự phản hồi. Dưới đây là cú pháp chung:

```
Accept-Charset: character_set [q=qvalue]
```

Nhiều bộ ký tự có thể được liệt kê riêng rẽ nhau bởi các dấu phảy và giá trị q tùy ý biểu diễn một mức độ chất lượng có thể chấp nhận cho các bộ ký tự không được ưa thích hơn trên một miền từ 0 đến 1. Dưới đây là ví dụ:

```
Accept-Charset: iso-8859-5, unicode-1-1; q=0.8
```

Giá trị đặc biệt "*", nếu có trong trường **Accept-Charset**, kết nối mọi bộ ký tự và nếu không có giá trị trường **Accept-Charset** nào, thì mặc định là bất kỳ bộ ký tự nào cũng có thể được chấp nhận.

Trường Accept-Encoding

Trường này tương tự như Accept, nhưng hạn chế mã hóa nội dung là có thể chấp nhận trong phản hồi. Cú pháp chung là:

```
Accept-Encoding: encoding types
```

Các ví dụ là như sau:

```
Accept-Encoding: compress, gzip Accept-Encoding: Accept-Encoding: * Accept-Encoding: compress;q=0.5, gzip;q=1.0 Accept-Encoding: gzip;q=1.0, identity; q=0.5, *;q=0
```

Trường Accept-Language

Trường này tương tự như Accept, nhưng hạn chế bộ thiết lập của các ngôn ngữ tự nhiên là được ưa thích hơn khi một phản hồi tới một yêu cầu. Cú pháp chung là:

```
Accept-Language: language [q=qvalue]
```

Nhiều ngôn ngữ có thể được liệt kê phân biệt nhau bởi dấu phảy và giá trị q tùy ý biểu diễn một mức độ chất lượng có thể chấp nhận cho các ngôn ngữ không được ưa thích hơn trên miền từ 0 tới 1. Dưới đây là một ví dụ:

```
Accept-Language: da, en-gb;q=0.8, en;q=0.7
```

Trường Authorization (Sự ủy quyền)

Giá trị trường **Authorization** bao gồm các sự ủy nhiệm mà chứa thông tin ủy quyền của một user agent cho phạm vi nguồn đang được yêu cầu. Cú pháp chung là:

```
Authorization : credentials
```

Định cấu hình HTTP/1.0 định nghĩa giản đồ ủy quyền CƠ BẢN, nơi mà tham số ủy quyền là một chuỗi của **tên sử dụng:mật khẩu** được mã hóa trong cơ sở 64 bit. Dưới đây là ví dụ:

```
Authorization: BASIC Z3Vlc3Q6Z3Vlc3QxMjM=
```

Giá trị đã giải mã là **guest:guest123**, trong đó **guest** là tài khoản người dùng và **guest123** là mật khẩu.

Trường Cookie

Giá trị trường **Cookie** chứa một cặp tên/giá trị của thông tin được lưu giữ cho URL đó. Dưới đây là cú pháp chung:

```
Cookie: name=value
```

Nhiều cookie có thể được xác định phân biệt nhau bởi các dấu chấm phảy ";" như sau:

```
Cookie: name1=value1;name2=value2;name3=value3
```

Trường Expect

Trường **Expect** được sử dụng để chỉ ra rằng một bộ thiết lập cụ thể của các hành vi Server được yêu cầu bởi Client. Cú pháp chung là:

```
Expect : 100-continue | expectation-extension
```

Nếu một Server nhận một yêu cầu chứa một trường Expect mà bao gồm một độ dãn mong đợi mà nó không hỗ trợ, nó phải phản hồi với trạng thái 417 (sự mong đợi thất bại).

Trường From

Trường **From** chứa một địa chỉ thư điện tử cho người sử dụng mà kiểm soát user agent. Dưới đây là một cú pháp đơn giản:

```
From: webmaster@w3.org
```

Trường này có thể được sử dụng cho nhập các mục đích và như là một phương tiện cho việc xác nhận nguồn của các yêu cầu không khả thi hoặc không muốn.

Trường Host

Trường **Host** được sử dụng để xác định Internet host và số hiệu cổng của nguồn được yêu cầu. Cú pháp chung là:

```
Host : "Host" ":" host [ ":" port ] ;
```

Một **Host** mà không có bất kỳ thông tin port nào ngụy ý là port mặc định, mà là 80. Ví dụ, một yêu cầu trên Server ban đầu cho *http://w3.org/pub/WWW/* sẽ là:

```
GET /pub/WWW/ HTTP/1.1 Host: www.w3.org
```

Trường If-Match

Trường **If-Match** được sử dụng trong một method để làm cho nó có điều kiện. Header này yêu cầu Server để biểu diễn method được yêu cầu chỉ khi giá trị được cung cấp trong thẻ này kết nối với các thẻ đối tượng được cung cấp được biểu diễn bởi **Etag**. Cú pháp chung là:

```
If-Match : entity-tag
```

Một dấu (*) kết nối với bất cứ đối tượng nào, và sự truyền tải tiếp tục chỉ khi đối tượng tồn tại. Dưới đây là các ví dụ có thể:

```
If-Match: "xyzzy" If-Match: "xyzzy", "r2d2xxxx", "c3piozzzz" If-Match: *
```

Nếu không có thể đối tượng nào kết nối, hoặc nếu (*) được cung cấp và không đối tượng hiện tại nào tồn tại, Server không được trình bày method được yêu cầu, và phải trả lại một phản hồi là 412 (điều kiện trước thất bại).

Trường If-Modified-Since

Trường này được sử dụng với một method để làm cho nó có điều kiện. Nếu URL được yêu cầu không được chỉnh sửa từ thời gian đã được xác định trong trường này, một đối tượng sẽ không được trả lại từ Server; thay vào đó, một phản hồi 304 (không được chỉnh sửa) sẽ được trả lại mà không có bất cứ phần thân thông báo nào. Cú pháp chung của If-Modified-Since là:

```
If-Modified-Since : HTTP-date
```

Một ví dụ của trường là:

```
If-Modified-Since: Sat, 29 Oct 1994 19:43:31 GMT
```

Nếu không có thẻ đối tượng nào kết nối với, hoặc nếu "*" được cung cấp và không đối tượng hiện tại nào tồn tại,, Server không được trình bày method được yêu cầu, và phải trả lại phản hồi 412 (điều kiện trước thất bại).

Trường If-None-Match

Trường này được sử dụng với một method để làm cho nó có điều kiện. Trường này yêu cầu Server trình bày method được yêu cầu chỉ khi một trong số giá trị đã cho của thẻ này kết nối với các thẻ đối tượng đã được cung cấp biểu diễn bởi **Etag**. Cú pháp chung là:

```
If-None-Match : entity-tag
```

Một dấu * kết nối với bất kỳ đối tượng nào, và sự truyền tải tiếp tục chỉ khi đối tượng không tồn tại. Dưới đây là các ví dụ có thể có:

```
If-None-Match: "xyzzy" If-None-Match: "xyzzy", "r2d2xxxx", "c3piozzzz" If-None-Match: *
```

Trường If-Range

Trường **If-Range** có thể được sử dụng với một GET có điều kiện để yêu cầu chỉ một phần của đối tượng mà đang bị thất lạc, nếu nó không được thay đổi, và toàn bộ đối tượng nếu nó được thay đổi. Cú pháp chung như sau:

```
If-Range : entity-tag | HTTP-date
```

Hoặc một thẻ đối tượng hoặc một dữ liệu có thể được sử dụng để xác minh đối tượng nội bộ đã nhận. Ví dụ:

```
If-Range: Sat, 29 Oct 1994 19:43:31 GMT
```

Tại đây, nếu tài liệu không được chỉnh sửa từ ngày đã cho, Server trả lại dãy byte được cung cấp bởi trường Range, nếu không thì nó trả lại tất cả các tài liệu mới.

Trường If-Unmodified-Since

Trường này được sử dụng với một method để làm cho nó có điều kiện. Cú pháp chung là:

```
If-Unmodified-Since : HTTP-date
```

Nếu nguồn được yêu cầu không được chỉnh sửa từ khi thời gian đã được xác định trong trường này, Server sẻ thực hiện hoạt động được yêu cầu nếu như If-Unmodified-Since không biểu diễn. Ví dụ:

```
If-Unmodified-Since: Sat, 29 Oct 1994 19:43:31 GMT
```

Nếu yêu cầu có kết quả là bất cứ gì khác ngoài một trạng thái là 2xx hoặc 4xx, thì trường **If-Unmodified-Since** nên được bỏ qua.

Trường Max-Forwards

Trường này cung cấp một kỹ thuật với các phương thức TRACE và OPTIONS để giới hạn số các trạm ủy quyền hoặc gateway mà có thể chuyển tiếp yêu cầu tới Server kế tiếp. Đây là cú pháp chung:

```
Max-Forwards : n
```

Giá trị Max-Forward là một số nguyên hệ thập phân chỉ rằng số lần còn lại của thông báo yêu cầu này có thể được chuyển tiếp. Điều này là hữu ích cho debug với phương thức TRACE, tránh được các vòng lặp vô hạn. Ví dụ:

```
Max-Forwards : 5
```

Trường này có thể bị bỏ qua với tất cả các phương thức được định nghĩa trong định cấu hình HTTP.

Trường Proxy-Authorization

Trường này cho phép Client xác nhận chính nó (hoặc người sử dụng của nó) tới một trạm ủy quyền mà yêu cầu sự ủy nhiệm. Đây là cú pháp chung:

```
Proxy-Authorization : credentials
```

Giá trị trường **Proxy-Authorization** bao gồm các sự ủy nhiệm chứa thông tin ủy nhiệm của user agent cho trạm ủy nhiệm và/hoặc phạm vi của nguồn được yêu cầu.

Trường Range

Trường **Range** xác định dãy nội bộ của nội dung được yêu cầu từ tài liệu. Cú pháp chung là:

```
Range: bytes-unit=first-byte-pos "-" [last-byte-pos]
```

Giá trị First-byte-pos trong một byte-range-spec chung cấp byte-offset của byte đầu tiên trong một dãy. Giá trị last-byte-pos cung cấp byte-offset của byte cuối cùng trong dãy; đó là, các vị trí byte được xác định. Bạn có thể xác định một đơn vị byte như các byte. Byte offset bắt đầu tại 0. Một số ví dụ đơn giản như sau:

```
- The first 500 bytes  Range: bytes=0-499  - The second 500 bytes Range:
bytes=500-999  - The final 500 bytes Range: bytes=-500  - The first and last
bytes only Range: bytes=0-0,-1
```

Nhiều dãy có thể được liệt kê, phân biệt nhau bởi dấu phảy. Nếu ký số đầu tiên trong dãy byte phân biệt nhau bởi dấu phảy bị bỏ quên, dãy được giả sử là tính toán từ phần cuối của tài liệu. Nếu ký số thứ hai bị bỏ quên, dãy là byte thứ n tới phần cuối tài liệu.

Trường Referer

Trường này cho phép Client xác định địa chỉ URI của nguồn mà từ đó URL đã được yêu cầu. Cú pháp chung là như sau:

```
Referer : absoluteURI | relativeURI
```

Dưới đây là ví dụ:

```
Referer: http://www.tutorialspoint.org../http/index.jsp
```

Nếu giá trị trường là một URI quan hệ, nó nên được phiên dịch liên quan tới *Request-URI*.

Trường TE

Trường này chỉ sự mở rộng nào mà *transfer-coding* đang muốn chấp nhận trong phản hồi và có hoặc không nó đang muốn chấp nhận các trường trailer trong một *transfer-coding*được đóng khối. Sau đây là cú pháp chung:

```
TE   : t-codings
```

Sự hiện diện của từ khóa “trailers” chỉ rằng Client đang muốn chấp nhận các trường trailer trong một transfer-coding được đóng khối và nó được xác định theo một trong 2 cách:

```
TE: deflate TE: TE: trailers, deflate;q=0.5
```

Nếu giá trị trường TE là trống hoặc không trường TE nào hiện diện, thì khi đó chỉ có *transfer-coding* được đóng khối (*chunked*). Một thông báo với không *transfer-coding* là luôn luôn có thể chấp nhận.

Trường User-Agent

Trường **User-Agent** chứa thông tin về tác nhân người sử dụng tạo yêu cầu. Sau đây là cú pháp chung:

```
User-Agent : product | comment
```

Ví dụ:

```
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
```

Các trường Phản hồi từ Server

Trường Accept-Ranges

Trường này cho phép Server chỉ sự chấp nhận của nó về các dãy yêu cầu cho một nguồn. Cú pháp chung là:

```
Accept-Ranges  : range-unit | none
```

Ví dụ, một Server mà chấp nhận các yêu cầu về dãy byte có thể gửi:

```
Accept-Ranges: bytes
```

Server mà không chấp nhận bất kỳ loại dãy yêu cầu cho một nguồn có thể gửi:

```
Accept-Ranges: none
```

Điều này sẽ khuyên Client không cố gắng để chiếm được một dãy yêu cầu.

Trường Age

Trường **Age** chuyển tính toán về số lượng thời gian từ khi phản hồi được tạo ra tại Server ban đầu của người gửi. Cú pháp chung là:

```
Age : delta-seconds
```

Giá trị Age là các số nguyên thập phân không phủ định, biểu diễn thời gian bằng giây. Sau đây là một ví dụ đơn giản:

```
Age: 1030
```

Một HTTP/1.1 Server mà bao gồm một bộ nhớ ẩn phải bao gồm một trường Age trong mỗi phản hồi được tạo từ bộ nhớ ẩn riêng của nó.

Trường ETag

Trường **Etag** cung cấp giá trị hiện tại của thẻ đối tượng cho biến thể được yêu cầu. Cú pháp chung là:

```
ETag :  entity-tag
```

Dưới đây là một số ví dụ đơn giản:

```
ETag: "xyzzy" ETag: W/"xyzzy" ETag: ""
```

Trường Location

Trường **Location** được sử dụng để điều hướng lại người nhận tới một vị trí khác ngoài Reqest-URI. Cú pháp chung là:

```
Location : absoluteURI
```

Dưới đây là một ví dụ đơn giản:

```
Location: http://www.tutorialspoint.org../http/index.jsp
```

Trường Content-Location khác Location ở trong đó mà Content-Location xác nhận vị trí ban đầu của đối tượng được bao gồm trong yêu cầu.

Trường Proxy-Authenticate

Trường này phải được bao gồm như là một phần của phản hồi 407. Cú pháp chung là:

```
Proxy-Authenticate  : challenge
```

Trường Retry-After

Trường này có thể được sử dụng với một phản hồi 503 (Service Unavailable - dịch vụ không có sẵn) để chỉ rằng dịch vụ được mong đợi để là không có sẵn trong vòng bao lâu tới Client đang yêu cầu. Cú pháp chung là:

```
Retry-After : HTTP-date | delta-seconds
```

Các ví dụ:

```
Retry-After: Fri, 31 Dec 1999 23:59:59 GMT Retry-After: 120
```

Trong ví dụ sau, sự trì hoãn là 2 phút.

Trường Server

Trường này chứa thông tin về phần mềm được sử dụng bởi Server ban đầu để kiểm soát yêu cầu. Cú pháp chung là:

```
Server : product | comment
```

Sau đây là ví dụ:

```
Server: Apache/2.2.14 (Win32)
```

Nếu phản hồi đang được chuyển tiếp qua một trạm ủy quyền, ứng dụng ủy quyền không được chỉnh sửa trường Server.

Trường Set-Cookie

Trường này chứa một cặp tên/giá trị của thông tin để giữ lại cho URL. Cú pháp chung là:

```
Set-Cookie: NAME=VALUE; OPTIONS
```

Trường phản hồi Set-Cookie bao gồm Set-Cookie dấu hiệu, được theo sau bởi một danh sách được phân biệt bằng dấu phẩy của một hoặc nhiều cookie. Ở đây là các giá trị có thể mà bạn có thể xác định như là các tính năng:

STT	Các tính năng và Miêu tả
1	**Comment=comment** Tính năng này có thể được sử dụng để xác định bất cứ lời bình nào liên kết với cookie.
2	**Domain=domain** Thuộc tính Domain xác định miền mà với nó cookie là có hiệu lực.
3	**Expires=Date-time** Ngày mà cookie sẽ hết hạn. Nếu nó là trống, cookie sẽ hết hạn khi khách sử dụng rời khỏi trình duyệt.
4	**Path=path** Thuộc tính path xác định bộ thiết lập phụ của các URL mà cookie này áp dụng.
5	**Secure** Nó chỉ dẫn user agent để trả lại cookie chỉ ở dưới dạng một kết nối an toàn.

Sau đây là một ví dụ của một cookie đơn giản được tạo bởi Server:

```
Set-Cookie: name1=value1,name2=value2; Expires=Wed, 09 Jun 2021 10:18:14 GMT
```

Trường Vary

Trường **Vary** xác định rằng đối tượng có nhiều nguồn và vì thế có thể theo nhiều cách để tới một danh sách yêu cầu đã được xác định. Sau đây là cú pháp chung:

```
Vary : field-name
```

Bạn có thể xác định nhiều Header phân biệt nhau bởi dấu phảy và một giá trị là dấu * mà không xác định các tham số (không giới hạn tới các Header yêu cầu). Sau đây là ví dụ đơn giản:

```
Vary: Accept-Language, Accept-Encoding
```

Ở đây, các tên trường là không nhạy cảm.

Trường WWW-Authenticate

Trường này phải được bao gồm trong các thông báo phản hồi 401 (không được ủy quyền). Giá trị trường bao gồm ít nhất một challenge (hiệu lệnh) mà chỉ dẫn giản đồ ủy quyền và các tham số có thể áp dụng tới URI yêu cầu. Cú pháp chung là:

```
WWW-Authenticate : challenge
```

Giá trị tường có thể chứa nhiều hơn một challenge, hoặc nếu nhiều hơn một trường WWW-Authenticate được cung cấp, các nội dung của chính challenge đó có thể chứa một danh sách được phân biệt bởi dấu phảy của các tham số ủy quyền. Sau đây là một ví dụ đơn giản:

```
WWW-Authenticate: BASIC realm="Admin"
```

Entity Headers

Trường Allow

Trường này liệt kê bộ thiết lập của các method được hỗ trợ bởi nguồn được xác nhận bởi Request-URI. Cú pháp chung là:

```
Allow : Method
```

Bạn có thể xác định nhiều phương thức được phân biệt bởi dấu phảy. Sau đây là một ví dụ đơn giản:

```
Allow: GET, HEAD, PUT
```

Trường này không ngăn cản một Client từ việc cố gắng thử các method khác.

Trường Content-Encoding

Trường này được sử dụng như một bộ chỉnh sửa tới kiểu phương tiện. Cú pháp chung là:

```
Content-Encoding : content-coding
```

Mã hóa nội dung là một thuộc tính của một đối tượng được xác định bởi Request-URI. Sau đây là một ví dụ đơn giản:

```
Content-Encoding: gzip
```

Nếu mã hóa nội dung của một đối tượng là một thông báo yêu cầu là không được chấp nhận bởi Server nguồn, Server sẽ phản hồi với một mã trạng thái 415 (kiểu phương tiện không được hỗ trợ).

Trường Content-Language

Trường này miêu tả các ngôn ngữ tự nhiên của người đọc đã dự định cho đối tượng đã bao gồm. Sau đây là cú pháp chung:

```
Content-Language : language-tag
```

Nhiều ngôn ngữ có thể được liệt kê cho nội dung mà được dự định cho nhiều người đọc. Sau đây là một ví dụ đơn giản:

```
Content-Language: mi, en
```

Mục đích đầu tiên của Content-Language là để cho phép một người sử dụng để xác nhận và tạo sự khác biệt các đối tượng theo ngôn ngữ được ưa thích hơn của riêng người dùng.

Trường Content-Length

Trường này chỉ dẫn kích cỡ của phần thân đối tượng, trong số thập phân của hệ 8, được gửi tới người nhận hoặc, trong trường hợp của phương thức HEAD, kích cỡ của phần thân đối tượng mà sẽ được gửi, có yêu cầu là một GET. Cú pháp chung là:

```
Content-Length : DIGITS
```

Sau đây là một ví dụ đơn giản:

```
Content-Length: 3495
```

Bất kỳ Conten-Length nào lớn hơn hoặc bằng là một giá trị có hiệu lực.

Trường Content-Location

Trường này có thể được sử dụng để cung cấp vị trí nguồn cho đối tượng được bao gồm trong thông báo khi đối tượng đó là có thể truy cập từ một vị trí riêng biệt từ một URI của nguồn được yêu cầu. Cú pháp chung là:

```
Content-Location:  absoluteURI | relativeURI
```

Sau đây là một ví dụ đơn giản:

```
Content-Location: http://www.tutorialspoint.org../http/index.jsp
```

Giá trị của Content-Location cũng định nghĩa URI cơ sở cho đối tượng.

Trường Content-MD5

Trường này có thể được sử dụng để cung cấp một hệ thống phân loại MD5 của đối tượng cho việc kiểm tra tính nguyên vẹn của thông tin tới người nhận. Cú pháp chung là:

```
Content-MD5  : md5-digest using base64 of 128 bit MD5 digest as per RFC 1864
```

Sau đây là một ví dụ đơn giản:

```
Content-MD5  : 8c2d46911f3f5a326455f0ed7a8ed3b3
```

Hệ thống phân loại MD5 được tính toán hóa dựa trên cơ sở nội dung của phần thân thực thể, bao gồm bất cứ mã hóa nội dung nào mà đã được áp dụng, nhưng không bao gồm bất cứ mã hóa truyền tải được áp dụng tới phần thân thông báo.

Trường Content-Range

Trường này được gửi với một phần thân thực thể nội bộ để xác định nơi trong toàn bộ phần thân đối tượng mà phần thân nội bộ nên được áp dụng. Cú pháp chung là:

```
Content-Range : bytes-unit SP first-byte-pos "-" last-byte-pos
```

Các ví dụ của các giá trị byte-content-range-spec, giả sử rằng đối tượng chứa một tổng của 1234 byte:

```
- The first 500 bytes: Content-Range : bytes 0-499/1234  - The second 500
bytes: Content-Range : bytes 500-999/1234  - All except for the first 500
```

```
bytes: Content-Range : bytes 500-1233/1234  - The last 500 bytes: Content-Range
: bytes 734-1233/1234
```

Khi một thông báo HTTP bao gồm nội dung của một dãy đơn, nội dung này được truyền tải với một Content-Range, và một Content-Length chỉ số byte được truyền tải thực sự. Ví dụ:

```
HTTP/1.1 206 Partial content Date: Wed, 15 Nov 1995 06:25:24 GMT Last-Modified:
Wed, 15 Nov 1995 04:58:08 GMT Content-Range: bytes 21010-47021/47022 Content-
Length: 26012 Content-Type: image/gif
```

Trường Content-Type

Trường này chỉ dẫn kiểu phương tiện của phần thân đối tượng được gửi tới người nhận hoặc, trong trường hợp phương thức HEAD, kiểu phương tiện mà sẽ được gửi, có yêu cầu là GET. Cú pháp chung là:

```
Content-Type : media-type
```

Sau đây là một ví dụ:

```
Content-Type: text/html; charset=ISO-8859-4
```

Trường Expires

Trường này cung cấp Ngày/Thời gian sau đó phản hồi được cho là hết hạn. Cú pháp chung:

```
Expires : HTTP-date
```

Sau đây là một ví dụ:

```
Expires: Thu, 01 Dec 1994 16:00:00 GMT
```

Trường Last-Modified

Trường này chỉ ngày và thời gian tại đó Server ban đầu tin rằng sự biến đổi được chỉnh sửa lần cuối. Cú pháp chung là:

```
Last-Modified: HTTP-date
```

Sau đây là một ví dụ:

```
Last-Modified: Tue, 15 Nov 1994 12:45:26 GMT
```