RFC 7234

Hypertext Transfer Protocol (HTTP/1.1): Caching

문서 최종 수정일	2020-05-26
원문 복사일	2020-05-06
번역 및 정리	이병록(roka88)
이메일	roka88.dev@gmail.com

PROPOSED STANDARD
Errata Exist

Internet Engineering Task Force (IETF)
Request for Comments: 7234
Obsoletes: 2616
Category: Standards Track
ISSN: 2070-1721

R. Fielding, Ed.
Adobe
M. Nottingham, Ed.
Akamai
J. Reschke, Ed.
greenbytes
June 2014

Hypertext Transfer Protocol (HTTP/1.1): Caching

Abstract

The Hypertext Transfer Protocol (HTTP) is a stateless application-level protocol for distributed, collaborative, hypertext information systems. This document defines HTTP caches and the associated header fields that control cache behavior or indicate cacheable response messages.

하이퍼텍스트 전송 프로토콜(HTTP)은 분산, 협업, 하이퍼텍스트 정보 시스템을 위한 상태 비저장 애플리케이션 레벨 프로토콜이다. 이 문서는 캐시 동작을 제어하거나 캐시 가능한 응답 메시지를 나타내는 HTTP 캐시 및 관련 헤더 필드를 정의한다.

Status of This Memo

This is an Internet Standards Track document.

이것은 인터넷 표준 추적 문서이다.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

이 문서는 Internet Engineering Task Force(IETF)의 제품이다. 문서는 IETF 공동체의 합의를 나타낸다. 문서는 공개 검토를 받아왔으며 Internet Engineering Starting Group (IESG)에 의해 발행 승인를 받았다. 인터넷 표준의 추가 정보는 RFC 5741 Section 2에서 확인할 수 있다.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7234.

이 문서에 대한 현재 상태 정보는 정오표와 피드백을 어떻게 제공하는 방법은 http://www.rfc-editor.org/info/rfc7234 에서 얻을 수 있다.

Copyright Notice

Table of Contents

Appendix C. Collected ABNF

1. Introduction

HTTP is typically used for distributed information systems, where performance can be improved by the use of response caches. This document defines aspects of HTTP/1.1 related to caching and reusing response messages.

HTTP는 일반적으로 분산 정보 시스템에 사용되며, 여기서 응답 캐시의 사용으로 성능이 향상 될 수 있다. 이 문서는 응답 메시지 캐싱 및 재사용과 관련된 HTTP/1.1의 측면을 정의한다.

An HTTP cache is a local store of response messages and the subsystem that controls storage, retrieval, and deletion of messages in it. A cache stores cacheable responses in order to reduce the response time and network bandwidth consumption on future, equivalent requests. Any client or server MAY employ a cache, though a cache cannot be used by a server that is acting as a tunnel.

HTTP 캐시는 응답 메시지의 로컬 저장소와 그 메시지의 저장, 검색 및 삭제를 제어하는 하위 시스템이다. 캐쉬는 향후 동등한 요청에 대한 응답 시간과 네트워크 대역폭 소비를 줄이기 위해 캐시 가능한 응답을 저장한다. 캐시는 터널 역할을 하는 서버에서 사용할 수 없지만, 클라이언트나 서버는 캐시를 사용할 수 있다.(MAY)

A shared cache is a cache that stores responses to be reused by more than one user; shared caches are usually (but not always) deployed as a part of an intermediary. A private cache, in contrast, is dedicated to a single user; often, they are deployed as a component of a user agent.

공유 캐시는 둘 이상의 사용자가 재사용할 응답을 저장하는 캐시로, 공유 캐시는 보통(항상 그렇지는 않지만) 중개자의 한 부분으로 배포된다. 반면, 개인 캐시는 단일 사용자 전용이며, 종종 사용자 에이전트의 구성 요소로 배포된다.

The goal of caching in HTTP/1.1 is to significantly improve performance by reusing a prior response message to satisfy a current request. A stored response is considered "fresh", as defined in Section 4.2, if the response can be reused without “validation" (checking with the origin server to see if the cached response remains valid for this request). A fresh response can therefore reduce both latency and

network overhead each time it is reused. When a cached response is not fresh, it might still be reusable if it can be freshened by validation (Section 4.3) or if the origin is unavailable (Section 4.2.4).

HTTP/1.1의 캐싱의 목표는 현재 요청을 충족시키기 위해 이전 응답 메시지를 재사용함으로써 성능을 크게 향상시키는 것이다. 저장된 응답은 Section 4.2에서 정의한 대로 "fresh(이하 신선한)" 것으로 간주되며, 응답은 “validation(이하 검토)” 없이 재사용될 수 있는 경우(이 요청에 대해 캐시된 응답이 유효한지 확인하기 위해 원서버에 확인). 따라서 새로운 응답은 재사용될 때마다 대기 시간과 네트워크 오버헤드를 모두 줄일 수 있다. 캐시된 응답이 신선하지 않은 경우에도 검토(Section 4.3)를 통해 신선하게 할 수 있거나 원본을 사용할 수 없는 경우에도 재사용할 수 있다.(Section 4.2.4)

1.1. Conformance and Error Handling

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT”, "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Conformance criteria and considerations regarding error handling are defined in Section 2.5 of [RFC7230].

1.2. Syntax Notation

This specification uses the Augmented Backus-Naur Form (ABNF) notation of [RFC5234] with a list extension, defined in Section 7 of [RFC7230], that allows for compact definition of comma-separated lists using a '#' operator (similar to how the '*' operator indicates repetition). Appendix B describes rules imported from other documents. Appendix C shows the collected grammar with all list operators expanded to standard ABNF notation.

1.2.1. Delta Seconds

The delta-seconds rule specifies a non-negative integer, representing time in seconds.

delta-seconds 규칙은 시간(초)을 나타내는 음수가 아닌 정수를 지정한다.

```
delta-seconds  = 1*DIGIT
```

A recipient parsing a delta-seconds value and converting it to binary form ought to use an arithmetic type of at least 31 bits of non-negative integer range. If a cache receives a delta-seconds value greater than the greatest integer it can represent, or if any of its subsequent calculations overflows, the cache MUST consider the value to be either 2147483648 (2^31) or the greatest positive integer it can conveniently represent.

delta-seconds 값을 구문 분석하여 이진 형식으로 변환하는 수신자는 음수가 아닌 정수 범위의 최소 31비트의 산술 유형을 사용해야 한다. 캐시가 나타낼 수 있는 최대 정수보다 큰 delta-seconds 값을 수신하거나 후속 계산이 오버플로되는 경우, 캐시는 값을 2147483648 (2^31) 또는 편리하게 나타낼 수 있는 최대 양의 정수 중 하나로 간주해야 한다.(MUST)

Note: The value 2147483648 is here for historical reasons, effectively represents infinity (over 68 years), and does not need to be stored in binary form; an implementation could produce it as a canned string if any overflow occurs, even if the calculations are performed with an arithmetic type incapable of directly representing that number. What matters here is that an overflow be detected and not treated as a negative value in later calculations.

참고: 2147483648 값의 역사적 이유는 여기에 있으며, 사실상 무한대(68년 이상)를 나타내며, 2진수 형식으로 저장할 필요가 없다; 그 계산이 그 숫자를 직접 나타낼 수 없는 산술 형식으로 수행되더라도, 오버플로가 발생한다면 구현은 그 값을 예정된 문자로 실행할 것이다. 여기서 중요한 것은 오버플로가 감지되어 이후의 계산에서 음의 값으로 취급되지 않는다는 것이다.

2. Overview of Cache Operation

Proper cache operation preserves the semantics of HTTP transfers ([RFC7231]) while eliminating the transfer of information already held in the cache. Although caching is an entirely OPTIONAL feature of HTTP, it can be assumed that reusing a cached response is desirable and that such reuse is the default behavior when no requirement or local configuration prevents it. Therefore, HTTP cache requirements are focused on preventing a cache from either storing a non-reusable response or reusing a stored response inappropriately, rather than mandating that caches always store and reuse particular responses.

적절한 캐시 동작은 이미 캐시에 저장된 정보의 전송을 제거하는 동시에 HTTP 전송의 의미론을([RFC7231]) 보존한다. 캐싱은 전적으로 HTTP의 선택적 기능이지만 캐시된 응답을 재사용하는 것이 바람직하며, 어떤 요구사항이나 로컬 구성으로도 이를 차단하지 못할 때 그러한 재사용은 기본 동작이라고 가정할 수 있다. 따라서, HTTP 캐시 요구사항은 캐시가 항상 특정 응답을 저장하고 재사용하도록 명령하기보다는, 캐시가 재사용할 수 없는 응답을 저장하거나 저장된 응답을 부적절하게 재사용하는 것을 방지하는 데 초점을 맞추고 있다.

Each cache entry consists of a cache key and one or more HTTP responses corresponding to prior requests that used the same key. The most common form of cache entry is a successful result of a retrieval request: i.e., a 200 (OK) response to a GET request, which contains a representation of the resource identified by the request target (Section 4.3.1 of [RFC7231]). However, it is also possible to cache permanent redirects, negative results (e.g., 404 (Not Found)), incomplete results (e.g., 206 (Partial Content)), and responses to methods other than GET if the method's definition allows such caching and defines something suitable for use as a cache key.

각 캐시 항목은 동일한 키를 사용한 이전 요청에 해당하는 캐시 키와 하나 이상의 HTTP 응답으로 구성된다. 캐시 항목의 가장 일반적인 형태는 검색 요청의 성공적인 결과, 즉 GET 요청에 대한 200 (OK) 응답으로, 요청 대상으로 식별된 리소스의 표현을 포함한다([RFC7231]의 Section 4.3.1). 그러나 영구 리다이렉트, 부정적 결과(e.g., 404 (Not Found)), 불완전한 결과(e.g., 206 (Partial Content)), 메서드의 정의가 그러한 캐싱을 허용하고 캐시 키로 사용하기에 적합한 것을 정의한다면 GET 이외의 메서드에 대한 대응도 가능하다.

The primary cache key consists of the request method and target URI. However, since HTTP caches in common use today are typically limited to caching responses to GET, many caches simply decline other methods and use only the URI as the primary cache key.

기본 캐시 키는 요청 메서드와 대상 URI로 구성된다. 그러나 오늘날 일반적으로 사용되는 HTTP 캐시는 일반적으로 GET에 대한 응답 캐싱에만 국한되므로, 많은 캐시는 다른 메서드를 간단히 거부하고 URI만 기본 캐시 키로 사용한다.

If a request target is subject to content negotiation, its cache entry might consist of multiple stored responses, each differentiated by a secondary key for the values of the original request's selecting header fields (Section 4.1).

요청 대상이 콘텐츠 협상 대상인 경우, 캐시 항목은 원래 요청의 선택 헤더 필드 값에 대한 보조 키로 각각 구분되는 복수의 저장된 응답으로 구성될 수 있다(Section 4.1).

3. Storing Responses in Caches

A cache MUST NOT store a response to any request, unless:

캐시는 다음이 아닌 한 모든 요청에 대한 응답을 저장해서는 안 된다.(MUST NOT)

o The request method is understood by the cache and defined as being cacheable, and

o 요청 메서드는 캐시에 의해 이해되고 캐시가 가능한 것으로 정의되며,

o the response status code is understood by the cache, and

o 응답 상태 코드는 캐시에 의해 이해되며,

o the "no-store" cache directive (see Section 5.2) does not appear in request or response header fields, and

o "no-store" 캐시 지시어(Section 5.2 참조)는 요청 또는 응답 헤더 필드에 없고

o the "private" response directive (see Section 5.2.2.6) does not appear in the response, if the cache is shared, and

o "private" 응답 지시어(Section 5.2.2.6 참조)는 캐시를 공유하는 경우, 응답 내에 없다.

o the Authorization header field (see Section 4.2 of [RFC7235]) does not appear in the request, if the cache is shared, unless the response explicitly allows it (see Section 3.2), and

o 응답에서 명시적으로 허락되지 않는 한, 캐시를 공유하는 경우([RFC7235]의 Section 4.2 참조) Authorization 헤더 필드가 요청 내에 없다.

o the response either:

o 다음 중 하나:

* contains an Expires header field (see Section 5.3), or

* Expires 헤더 필드 포함(Section 5.3 참조), 또는

* contains a max-age response directive (see Section 5.2.2.8), or

* max-age 응답 지시어를 포함(Section 5.2.2.8 참조), 또는

* contains a s-maxage response directive (see Section 5.2.2.9) and the cache is shared, or

* s-maxage 응답 지시어(Section 5.2.2.9 참조)를 포함하며, 캐시를 공유하거나, 또는

* contains a Cache Control Extension (see Section 5.2.3) that allows it to be cached, or

* Cache 제어 확장 기능(Section 5.2.3 참조)을 포함하거나, 또는

* has a status code that is defined as cacheable by default (see Section 4.2.2), or

* 기본적으로 캐시 가능으로 정의되는 상태 코드(Section 4.2.2 참조), 또는

* contains a public response directive (see Section 5.2.2.5).

* 범용 응답 지시어를 포함한다(Section 5.2.2.5 참조).

Note that any of the requirements listed above can be overridden by a cache-control extension; see Section 5.2.3.

위에 나열된 요구 사항은 cache-control 확장에 의해 재정의될 수 있다는 점에 참조한다. Section 5.2.3을 참조한다.

In this context, a cache has "understood" a request method or a response status code if it recognizes it and implements all specified caching-related behavior.

이러한 맥락에서 캐시는 지정된 모든 캐싱 관련 동작을 인식하고 구현할 경우, 요청 메서드나 응답 상태 코드를 "understood(이하 이해)"한다고 한다.

Note that, in normal operation, some caches will not store a response that has neither a cache validator nor an explicit expiration time, as such responses are not usually useful to store. However, caches are not prohibited from storing such responses.

정상 동작 시, 일부 캐시는 캐시 검증자 또는 명시적 만료 시간이 없는 응답을 저장하지 않는다. 이러한 응답은 일반적으로 저장하기에 유용하지 않기 때문이다. 그러나 캐시는 이러한 응답을 저장하는 것을 막지 않는다.

3.1. Storing Incomplete Responses

A response message is considered complete when all of the octets indicated by the message framing ([RFC7230]) are received prior to the connection being closed. If the request method is GET, the response status code is 200 (OK), and the entire response header section has been received, a cache MAY store an incomplete response message body if the cache entry is recorded as incomplete. Likewise, a 206 (Partial Content) response MAY be stored as if it were an incomplete 200 (OK) cache entry. However, a cache MUST NOT store incomplete or partial-content responses if it does not support the Range and Content-Range header fields or if it does not understand the range units used in those fields.

커넥션이 닫히기 전에, 메시지 프레임([RFC7230])으로 표시된 모든 octet이 수신되면 응답 메시지가 완료된 것으로 간주된다. 요청 메서드가 GET이고 응답 상태 코드가 200(OK)이며,

응답 헤더 부문 전체가 수신된 경우, 캐시 항목이 불완전하다고 기록될 경우, 불완전한 캐시 응답 메시지 본문을 저장할 것이다. 마찬가지로, 206 (Partial Content) 응답은 불완전한 200 (OK) 캐시 항목인 것처럼 저장될 수 있다. 그러나 캐시가 Range 및 Content-Range 헤더 필드를 지원하지 않거나 해당 필드에 사용된 범위 단위를 이해하지 못하는 경우, 불완전하거나 partial-content 응답을 저장해서는 안 된다.(MUST NOT)

A cache MAY complete a stored incomplete response by making a subsequent range request ([RFC7233]) and combining the successful response with the stored entry, as defined in Section 3.3. A cache MUST NOT use an incomplete response to answer requests unless the response has been made complete or the request is partial and specifies a range that is wholly within the incomplete response. A cache MUST NOT send a partial response to a client without explicitly marking it as such using the 206 (Partial Content) status code.

캐시는 Section 3.3에 정의된 대로 후속 범위 요청([RFC7233])을 수행하고 성공적인 응답을 저장된 항목과 결합하여 저장된 불완전한 응답을 완료할 수 있다.(MAY) 캐시는 응답이 완료되지 않았거나 요청이 부분적이며 완전하게 불완전한 응답 범위 내에 있는 범위를 지정하지 않는 한, 요청에 응답하는 데 불완전한 응답을 사용해서는 안 된다.(MUST NOT) 캐시는 206 (Partial Content) 상태 코드를 사용하는 명시적인 표시 없이 클라이언트에 부분 응답을 보내서는 안 된다.(MUST NOT)

3.2. Storing Responses to Authenticated Requests

A shared cache MUST NOT use a cached response to a request with an Authorization header field (Section 4.2 of [RFC7235]) to satisfy any subsequent request unless a cache directive that allows such responses to be stored is present in the response.

공유 캐시는 해당 응답을 저장할 수 있는 캐시 지시어가 응답에 없는 한, 후속 요청을 충족하기 위해 Authorization 헤더 필드가 있는 요청에 대해 캐시된 응답을 사용해서는 안 된다.(MUST NOT) ([RFC7235] Section 4.2).

In this specification, the following Cache-Control response directives (Section 5.2.2) have such an effect: must-revalidate, public, and s-maxage.

이 명세에서 다음과 같은 Cache-Control 응답 지시어(Section 5.2.2)는 영향을 가진다: must-revalidate, public, 및 s-maxage

Note that cached responses that contain the "must-revalidate" and/or "s-maxage" response directives are not allowed to be served stale (Section 4.2.4) by shared caches. In particular, a response with either "max-age=0, must-revalidate" or "s-maxage=0" cannot be used to satisfy a subsequent request without revalidating it on the origin server.

"must-revalidate" 및/또는 "s-maxage" 응답 지시어가 포함된 캐시된 응답은 공유 캐시에 의해 오래된 것으로 제공되지 않는다(Section 4.2.4). 특히, "max-age=0, must-revalidate" 또는 "s-maxage=0" 중 하나를 사용한 응답은 원서버에서 재검토하지 않고 후속 요청을 충족시키는 데 사용할 수 없다.

3.3. Combining Partial Content

A response might transfer only a partial representation if the connection closed prematurely or if the request used one or more Range specifiers ([RFC7233]). After several such transfers, a cache might have received several ranges of the same representation. A cache MAY combine these ranges into a single stored response, and reuse that response to satisfy later requests, if they all share the same strong validator and the cache complies with the client requirements in Section 4.3 of [RFC7233].

커넥션이 일찍 닫히거나 요청이 하나 이상의 Range 지정자를 사용한 경우([RFC7233]) 응답은 부분 표현만 전송할 수 있다. 이러한 전송이 여러 번 이루어진 후, 캐시는 동일한 표현의 여러 범위를 수신했을 수 있다. 캐시는 이러한 범위를 단일 저장 응답으로 결합할 수 있으며,(MAY) 모두 동일한 강한 검증자를 공유하며 캐시가 [RFC7233]의 Section 4.3의 클라이언트 요구 사항을 준수하는 경우, 그 응답을 이후 충족하기 위해 재사용할 수 있다.

When combining the new response with one or more stored responses, a cache MUST:

새 응답을 하나 이상의 저장된 응답과 결합할 때 캐시는 다음을 수행해야 한다.(MUST)

o delete any Warning header fields in the stored response with warn-code 1xx (see Section 5.5);

o warn-code 1xx와 저장된 응답의 모든 Warning 헤더 필드를 삭제한다(Section 5.5 참조).

o retain any Warning header fields in the stored response with warn-code 2xx; and,

o warn-code 2xx와 저장된 응답의 모든 Warning 헤더 필드를 유지하며,

o use other header fields provided in the new response, aside from Content-Range, to replace all instances of the corresponding header fields in the stored response.

o 새 응답에 제공된 다른 헤더 필드를 사용하여, Content-Range 외에, 저장된 응답에 있는 해당 헤더 필드의 모든 인스턴스를 대체.

4. Constructing Responses from Caches

When presented with a request, a cache MUST NOT reuse a stored response, unless:

제시된 요청 시, 캐시는 저장된 응답을 재사용해서는 안 된다. 단, 다음과 같은 경우는 예외로 한다.

o The presented effective request URI (Section 5.5 of [RFC7230]) and that of the stored response match, and

o 제시된 유효한 요청 URI([RFC7230]의 Section 5.5)와 저장된 응답 일치하며,

o the request method associated with the stored response allows it to be used for the presented request, and

o 저장된 응답과 관련된 요청 메서드는 제시된 요청에 사용할 수 있도록 허용하며,

o selecting header fields nominated by the stored response (if any) match those presented (see Section 4.1), and

o 저장된 응답(있는 경우)에 의해 정해진 선택 헤더 필드가 제시된 헤더 필드와 일치하며 (Section 4.1 참조)

o the presented request does not contain the no-cache pragma (Section 5.4), nor the no-cache cache directive (Section 5.2.1), unless the stored response is successfully validated (Section 4.3), and

o 저장된 응답이 성공적으로 검증되지 않는 한(Section 4.3), 제시된 요청은 no-cache pragma(Section 5.4), no-cache 지시어(Section 5.2.1)는 포함되지 않으며,

o the stored response does not contain the no-cache cache directive (Section 5.2.2.2), unless it is successfully validated (Section 4.3), and

o 성공적으로 검증되지 않는 한(Section 4.3) 저장된 응답에는 no-cache 캐시 지시어 (Section 5.2.2)가 포함되지 않는다.

o the stored response is either:

o 저장된 응답은 다음 중 하나이다.

* fresh (see Section 4.2), or

* 신선한(Section 4.2 참조), 또는

* allowed to be served stale (see Section 4.2.4), or

* 오래된 것으로 제공됨(Section 4.2.4 참조), 또는

* successfully validated (see Section 4.3).

* 성공적으로 검증됨(Section 4.3 참조).

Note that any of the requirements listed above can be overridden by a cache-control extension; see Section 5.2.3.

위에 나열된 요구 사항은 cache-control 확장에 의해 재정의될 수 있다는 점에 유의한다. Section 5.2.3을 참조한다.

When a stored response is used to satisfy a request without validation, a cache MUST generate an Age header field (Section 5.1), replacing any present in the response with a value equal to the stored response's current_age; see Section 4.2.3.

저장된 응답을 사용하여 검토하지 않고 요청을 충족할 경우, 캐시는 Age 헤더 필드(Section 5.1)를 생성해야 하며,(MUST) 응답에 존재하는 모든 값을 저장된 응답의 current_age과 동일한 값으로 대체한다. Section 4.2.3을 참조한다.

A cache MUST write through requests with methods that are unsafe (Section 4.2.1 of [RFC7231]) to the origin server; i.e., a cache is not allowed to generate a reply to such a request before having forwarded the request and having received a corresponding response.

캐시는 안전하지 않은 메서드([RFC7231]의 Section 4.2.1)를 사용하여 요청을 원서버에 작성해야 한다.(MUST) 즉, 캐시는 요청을 전달하고 해당 응답을 수신하기 전에 해당 요청에 대한 응답을 생성할 수 없다.

Also, note that unsafe requests might invalidate already-stored responses; see Section 4.4.

또한, 안전하지 않은 요청은 이미 저장된 응답을 무효화할 수 있다는 점에 유의한다. Section 4.4를 참조한다.

When more than one suitable response is stored, a cache MUST use the most recent response (as determined by the Date header field). It can also forward the request with "Cache-Control: max-age=0" or "Cache-Control: no-cache" to disambiguate which response to use.

둘 이상의 적절한 응답을 저장할 때, 캐시는 (Date 헤더 필드에서 결정한) 최신 응답을 사용해야 한다. 또한 어느 응답에서 캐시를 사용해야 하는지 명확하게 하기 위해 "Cache-Control: max-age=0" 또는 "Cache-Control: no-cache"와 같이 요청을 전달할 수 있다.

A cache that does not have a clock available MUST NOT use stored responses without revalidating them upon every use.

사용 가능한 시계가 없는 캐시는 사용 시마다 저장된 응답을 다시 검증하지 않고 사용해서는 안 된다.(MUST NOT)

4.1. Calculating Secondary Keys with Vary

When a cache receives a request that can be satisfied by a stored response that has a Vary header field (Section 7.1.4 of [RFC7231]), it MUST NOT use that response unless all of the selecting header fields nominated by the Vary header field match in both the original request (i.e., that associated with the stored response), and the presented request.

캐시가 Vary 헤더 필드를 가지고 있는 저장된 응답으로 충족될 수 있는 요청을 수신할 때([RFC7231]의 Section 7.1.4), 원본 요청(즉, 저장된 응답과 관련된 요청)과 제시된 요청 모두에서 Vary 헤더 필드에 의해 지명된 모든 선택 헤더 필드가 일치하지 않는 한, 캐시는 저장된 응답을 사용해서는 안 된다.(MUST NOT)

The selecting header fields from two requests are defined to match if and only if those in the first request can be transformed to those in the second request by applying any of the following:

두 요청 중 선택 헤더 필드는 다음 중 하나를 적용하여 첫 번째 요청의 헤더 필드를 두 번째 요청의 헤더 필드로 변환할 수 있는 경우에만 일치하도록 정의된다.

o adding or removing whitespace, where allowed in the header field's syntax

o 헤더 필드의 구문에 허용되는 공백 추가 또는 제거

o combining multiple header fields with the same field name (see Section 3.2 of [RFC7230])

o 여러 헤더 필드를 동일한 필드 이름으로 결합([RFC7230]의 Section 3.2 참조)

o normalizing both header field values in a way that is known to have identical semantics, according to the header field's specification (e.g., reordering field values

when order is not significant; case-normalization, where values are defined to be case-insensitive)

o 헤더 필드 명세에 따라 동일한 의미론을 갖는 것으로 알려진 방식으로 두 헤더 필드 값을 정규화(e.g., 순서가 유의하지 않을 때 필드 값 순서 변경, 대/소문자 구분으로 정의된 경우)

If (after any normalization that might take place) a header field is absent from a request, it can only match another request if it is also absent there.

헤더 필드가 요청에 없는 경우(정규화 후) 헤더 필드가 없는 경우에만 다른 요청과 일치할 수 있다.

A Vary header field-value of "*" always fails to match.

Vary 헤더 field-value "*" 은 항상 일치하지 않는다.

The stored response with matching selecting header fields is known as the selected response.

선택 헤더 필드가 일치하는 저장된 응답을 선택된 응답으로 알려져있다.

If multiple selected responses are available (potentially including responses without a Vary header field), the cache will need to choose one to use. When a selecting header field has a known mechanism for doing so (e.g., qvalues on Accept and similar request header fields), that mechanism MAY be used to select preferred responses; of the remainder, the most recent response (as determined by the Date header field) is used, as per Section 4.

복수의 선택된 응답을 이용할 수 있는 경우(잠재적으로 Vary 헤더 필드가 없는 응답을 포함), 캐시는 사용할 응답을 선택해야 한다. 헤더 필드의 선택 메커니즘이 알려진 경우(예: Accept 및 유사한 요청 헤더 필드의 qvalue) 해당 메커니즘을 사용하여 선호된 응답을 선택할 수 있으며,(MAY) 나머지 부분 중 (Date 헤더 필드에서 결정한) 가장 최근의 응답이 Section 4에 따라 사용된다.

If no selected response is available, the cache cannot satisfy the presented request. Typically, it is forwarded to the origin server in a (possibly conditional; see Section 4.3) request.

선택된 응답을 사용할 수 없는 경우, 캐시는 제시된 요청을 충족할 수 없다. 일반적으로, (가능한 조건; Section 4.3 참조) 그것은 요청내에 원서버로 전달된다.

4.2. Freshness

A fresh response is one whose age has not yet exceeded its freshness lifetime. Conversely, a stale response is one where it has.

신선한 응답은 나이가 아직 신선도 수명을 초과하지 않은 것이다. 반대로, 오래된 응답은 그렇지 않은 것이다.

A response's freshness lifetime is the length of time between its generation by the origin server and its expiration time. An explicit expiration time is the time at which the origin server intends that a stored response can no longer be used by a cache without further validation, whereas a heuristic expiration time is assigned by a cache when no explicit expiration time is available.

응답의 신선도 수명은 원서버에 의한 생성과 만료 시간 사이의 시간이다. 명시적 만료 시간은 추가 검토 없이 저장된 응답을 캐시에 더 이상 사용할 수 없다고 원서버가 의도하는 시간인 반면, 휴리스틱 만료 시간은 명시적 만료 시간이 없을 때 캐시에 의해 할당된다.

A response's age is the time that has passed since it was generated by, or successfully validated with, the origin server.

응답의 나이는 원서버에 의해 생성되거나 성공적으로 유효성 검사에 성공한 이후 경과한 시간이다.

When a response is "fresh" in the cache, it can be used to satisfy subsequent requests without contacting the origin server, thereby improving efficiency.

캐시에서 응답이 "fresh"일 때, 원서버에 접속하지 않고 후속 요청을 충족시키는 데 사용할 수 있어 효율성이 향상된다.

The primary mechanism for determining freshness is for an origin server to provide an explicit expiration time in the future, using either the Expires header field (Section 5.3) or the max-age response directive (Section 5.2.2.8). Generally, origin servers will assign future explicit expiration times to responses in the belief that the representation is not likely to change in a semantically significant way before the expiration time is reached.

신선도를 결정하는 기본 메커니즘은 Expires 헤더 필드(Section 5.3) 또는 max-age 응답 지시어(Section 5.2.2.8)를 사용하여 원서버가 향후 명시적인 만료 시간을 제공하는 것이다. 일반적으로, 원서버는 표현이 만료 시간에 도달하기 전에 의미론적으로 중요한 방식으로 변경될 가능성이 낮다는 신뢰로 응답에 미래의 명시적 만료 시간을 할당한다.

If an origin server wishes to force a cache to validate every request, it can assign an explicit expiration time in the past to indicate that the response is already stale. Compliant caches will normally validate a stale cached response before reusing it for subsequent requests (see Section 4.2.4).

원서버가 모든 요청을 확인하기 위해 캐시를 강제로 실행하려는 경우, 응답은 이미 오래되었다는 것을 나타내기 위해 과거의 명시적인 만료 시간을 할당할 수 있다. 준수하는 캐시는 일반적으로 후속 요청에 다시 사용하기 전에 오래된 캐시된 응답의 유효성을 검사한다(Section 4.2.4 참조).

Since origin servers do not always provide explicit expiration times, caches are also allowed to use a heuristic to determine an expiration time under certain circumstances (see Section 4.2.2).

원서버가 항상 명시적인 만료 시간을 제공하는 것은 아니기 때문에 캐시는 특정 상황에서 만료 시간을 결정하기 위해 휴리스틱을 사용하는 것도 허용된다(Section 4.2.2 참조).

The calculation to determine if a response is fresh is:
응답이 신선한지 여부를 결정하는 계산은 다음과 같다.

```
response_is_fresh = (freshness_lifetime > current_age)
```

freshness_lifetime is defined in Section 4.2.1; current_age is defined in Section 4.2.3.

freshness_lifetime은 Section 4.2.1에 정의되어 있으며, current_age는 Section 4.2.3에 정의되어 있다.

Clients can send the max-age or min-fresh cache directives in a request to constrain or relax freshness calculations for the corresponding response (Section 5.2.1).

클라이언트는 해당 응답에 대한 신선도 계산을 제한하거나 완화하기 위한 요청으로 max-age 또는 min-fresh 캐시 지시어를 전송할 수 있다.(Section 5.2.1)

When calculating freshness, to avoid common problems in date parsing:

신선도를 계산할 때 날짜 구문 분석의 일반적인 문제를 방지하려면:

o Although all date formats are specified to be case-sensitive, a cache recipient SHOULD match day, week, and time-zone names case-insensitively.

o 모든 날짜 형식이 대소문자를 구분하도록 지정되어 있지만 캐시 수신자는 대소문자를 구분하지 않고 일, 주 및 타임존 이름과 일치해야 한다.(SHOULD)

o If a cache recipient's internal implementation of time has less resolution than the value of an HTTP-date, the recipient MUST internally represent a parsed Expires date as the nearest time equal to or earlier than the received value.

o 캐시 수신자의 시간 내부 구현이 HTTP-date 값보다 낮은 해상도를 가진 경우, 수신자는 내부적으로 구문 분석 Expires 날짜를 수신된 값과 같거나 이전인 가장 가까운 시간으로 나타내야 한다.(MUST)

o A cache recipient MUST NOT allow local time zones to influence the calculation or comparison of an age or expiration time.

o 캐시 수신자는 지역 시간대가 나이 또는 만료 시간의 계산 또는 비교에 영향을 미치지 않도록 해서는 안 된다.(MUST NOT)

o A cache recipient SHOULD consider a date with a zone abbreviation other than GMT or UTC to be invalid for calculating expiration.

o 캐시 수신자는 GMT 또는 UTC 이외의 zone 약자가 있는 날짜가 만료일 계산에 유효하지 않은 것으로 간주해야 한다.(SHOULD)

Note that freshness applies only to cache operation; it cannot be used to force a user agent to refresh its display or reload a resource. See Section 6 for an explanation of the difference between caches and history mechanisms.

신선도는 캐시 동작에만 적용되며, 사용자 에이전트에서 디스플레이를 새로 고치거나 리소스를 다시 로드하는 데 사용할 수 없다는 점에 유의한다. 캐시와 역사적 메커니즘의 차이에 대한 설명은 Section 6을 참조한다.

4.2.1. Calculating Freshness Lifetime

A cache can calculate the freshness lifetime (denoted as freshness_lifetime) of a response by using the first match of the following:

캐시는 다음 중 첫 번째 일치를 사용하여 응답의 신선도 수명(freshness_lifetime)을 계산할 수 있다:

o If the cache is shared and the s-maxage response directive (Section 5.2.2.9) is present, use its value, or

o 캐시가 공유되고 s-maxage 응답 지시어(Section 5.2.2.9)이 있는 경우, 해당 값을 사용하거나, 또는,

o If the max-age response directive (Section 5.2.2.8) is present, use its value, or

o max-age 응답 지시어(Section 5.2.2.8)이 있는 경우, 해당 값을 사용하거나, 또는,

o If the Expires response header field (Section 5.3) is present, use its value minus the value of the Date response header field, or

o Expires 응답 헤더 필드(Section 5.3)가 있는 경우 해당 값을 Date 응답 헤더 필드의 값을 뺀 값으로 사용하거나, 또는,

o Otherwise, no explicit expiration time is present in the response. A heuristic freshness lifetime might be applicable; see Section 4.2.2.

o 그렇지 않으면, 응답에 명시적인 만료 시간이 존재하지 않는다. 휴리스틱 신선도 수명이 적용될 수 있다. Section 4.2.2를 참조한다.

Note that this calculation is not vulnerable to clock skew, since all of the information comes from the origin server.

모든 정보는 원서버에서 제공되므로, 이 계산은 시계 왜곡에 취약하지 않는 것에 유의한다.

When there is more than one value present for a given directive (e.g., two Expires header fields, multiple Cache-Control: max-age directives), the directive's value is considered invalid. Caches are encouraged to consider responses that have invalid freshness information to be stale.

지정된 지시어에 대한 값이 두 개 이상 있을 때(e.g., 두 개의 Expires, 여러개의 Cache-Control: max-age 지시어) 지시어의 값은 유효하지 않은 것으로 간주된다. 캐시는 신선도 정보가 유효하지 않은 응답을 오래된 것으로 간주하도록 권장된다.

4.2.2. Calculating Heuristic Freshness

Since origin servers do not always provide explicit expiration times, a cache MAY assign a heuristic expiration time when an explicit time is not specified, employing algorithms that use other header field values (such as the Last-Modified time) to estimate a plausible expiration time. This specification does not provide specific algorithms, but does impose worst-case constraints on their results.

원서버가 항상 명시적 만료 시간을 제공하는 것은 아니기 때문에, 캐시는 다른 헤더 필드 값(Last-Modified 시간 등)을 사용하여 타당할 것 같은 만료 시간을 추정하는 알고리즘을 채택하여, 명시적 시간이 지정되지 않을 때 휴리스틱 만료 시간을 할당할 수 있다. 이 명세는 특정한 알고리즘을 제공하지는 않지만, 그 결과에 최악의 조건을 부과한다.

A cache MUST NOT use heuristics to determine freshness when an explicit expiration time is present in the stored response. Because of the requirements in Section 3, this means that, effectively, heuristics can only be used on responses without explicit freshness whose status codes are defined as cacheable by default (see Section 6.1 of [RFC7231]), and those responses without explicit freshness that have been marked as explicitly cacheable (e.g., with a “public" response directive).

캐시는 저장된 응답에 명시적 만료 시간이 있을 때 휴리스틱스를 사용하여 신선도를 결정해서는 안 된다.(MUST NOT) Section 3의 요구사항 때문에, 이 의미는, 효과적으로, 휴리스틱스는 상태 코드가 기본적으로 캐시 가능으로 정의된 명시적 신선도([RFC7231]의 Section 6.1 참조) 없는 응답과, 명시적으로 캐시 가능(e.g., "public" 응답 지시어와 함께)으로 표시된 명시적 신선도 없는 응답에만 사용할 수 있음을 의미한다.

If the response has a Last-Modified header field (Section 2.2 of [RFC7232]), caches are encouraged to use a heuristic expiration value that is no more than some fraction of the interval since that time. A typical setting of this fraction might be 10%.

응답에 Last-Modified 헤더 필드가 있는 경우([RFC7232]의 Section 2.2), 캐시는 그 시간 이후 간격의 일부 이하인 휴리스틱 만료 값을 사용하는 것이 좋다. 이 부분의 일반적인 설정은 10%일 수 있다.

When a heuristic is used to calculate freshness lifetime, a cache SHOULD generate a Warning header field with a 113 warn-code (see Section 5.5.4) in the response if its current_age is more than 24 hours and such a warning is not already present.

신선도 수명을 계산하기 위해 휴리스틱을 사용하는 경우, current_age가 24시간 이상이고 그러한 경고가 아직 존재하지 않는 경우, 캐시는 응답에 113 warn-code(Section 5.5.4 참조)를 가진 Warning 헤더 필드를 생성해야 한다.(SHOULD)

> Note: Section 13.9 of [RFC2616] prohibited caches from calculating heuristic freshness for URIs with query components (i.e., those containing '?'). In practice, this has not been widely implemented. Therefore, origin servers are encouraged to send explicit directives (e.g., Cache-Control: no-cache) if they wish to preclude caching.
>
> 참고: [RFC2616] Section 13.9에서는 캐시가 쿼리 구성요소가 있는 URI(즉, '?'가 포함된)에 대해 휴리스틱 신선도를 계산하는 것을 금지했다. 실제로 이는 널리 구현되지 않았다. 따라서 캐싱을 배제하고자 하는 경우 원서버는 명시적 지시어(예: Cache-Control: no-cache)를 전송하도록 권장된다.

4.2.3. Calculating Age

The Age header field is used to convey an estimated age of the response message when obtained from a cache. The Age field value is the cache's estimate of the number of seconds since the response was generated or validated by the origin server. In essence, the Age value is the sum of the time that the response has been resident in each of the caches along the path from the origin server, plus the amount of time it has been in transit along network paths.

Age 헤더 필드는 캐시에서 얻을 때 응답 메시지의 예상 나이를 전달하는 데 사용된다. Age 필드 값은 원서버에서 응답을 생성하거나 유효성을 검사한 이후(초)에 대한 캐시의 추정치이다. 본질적으로 Age 값은 응답이 원서버의 경로를 따라 각 캐시에 상주한 시간의 합과 네트워크 경로를 따라 전송된 시간의 합이다.

The following data is used for the age calculation:

다음의 데이터는 나이 계산을 위해 사용되었다:

age_value

The term "age_value" denotes the value of the Age header field (Section 5.1), in a form appropriate for arithmetic operation; or 0, if not available.

"age_value"라는 용어는 Age 헤더 필드(Section 5.1)의 값을 산술 연산에 적합한 형식으로 표시하거나, 사용할 수 없는 경우 0을 나타낸다.

date_value

The term "date_value" denotes the value of the Date header field, in a form appropriate for arithmetic operations. See Section 7.1.1.2 of [RFC7231] for the definition of the Date header field, and for requirements regarding responses without it.

"date_value"라는 용어는 Date 헤더 필드의 값을 산술 연산에 적합한 형식으로 나타낸다. Date 헤더 필드의 정의 및 Date 헤더 필드 없는 응답에 관한 요구사항은 [RFC7231] Section7.1.1.2을 참조한다.

now

The term "now" means "the current value of the clock at the host performing the calculation". A host ought to use NTP ([RFC5905]) or some similar protocol to synchronize its clocks to Coordinated Universal Time.

"now"라는 용어는 "계산을 수행하는 호스트 시계의 현재 값"을 말한다. 호스트는 NTP([RFC5905]) 또는 이와 유사한 프로토콜을 사용하여 시계를 UTC로 동기화해야 한다.

request_time

The current value of the clock at the host at the time the request resulting in the stored response was made.

저장된 응답에서 요청의 결과가 수행된 시점의 호스트에서 시계의 현재 값.

response_time

The current value of the clock at the host at the time the response was received.

응답을 수신한 시점의 호스트에서 시계의 현재 값.

A response's age can be calculated in two entirely independent ways:

응답의 나이는 완전히 독립적인 두 가지 방법으로 계산할 수 있다.

1. the "apparent_age": response_time minus date_value, if the local clock is reasonably well synchronized to the origin server's clock. If the result is negative, the result is replaced by zero.

1. "apparent_age": response_time - date_value, 로컬 시계가 원서버의 시계와 합리적으로 잘 동기화된 경우. 결과가 음수일 경우 결과는 0으로 대체된다.

2. the "corrected_age_value", if all of the caches along the response path implement HTTP/1.1. A cache MUST interpret this value relative to the time the request was initiated, not the time that the response was received.

2. "corrected_age_value"는, 응답 경로를 따라 모든 캐시가 HTTP/1.1을 구현하는 경우. 캐시는 응답을 받은 시간이 아니라 요청이 시작된 시간에 비례하여 이 값을 해석해야 한다.

```
apparent_age = max(0, response_time - date_value);

response_delay = response_time - request_time;
corrected_age_value = age_value + response_delay;
```

These are combined as

이 것들을 결합하면

```
corrected_initial_age = max(apparent_age, corrected_age_value);
```

unless the cache is confident in the value of the Age header field (e.g., because there are no HTTP/1.0 hops in the Via header field), in which case the corrected_age_value MAY be used as the corrected_initial_age.

캐시가 Age 헤더 필드의 값(e.g., Via 헤더 필드에 HTTP/1.0 홉이 없기 때문에)에 자신 있는 경우가 아니라면, corrected_age_value는 corrected_initial_age로 사용할 수 있다.(MAY)

The current_age of a stored response can then be calculated by adding the amount of time (in seconds) since the stored response was last validated by the origin server to the corrected_initial_age.

그러면 저장된 응답의 current_age은 원서버에서 마지막으로 검토 한 이후의 시간(초)을 corrects_initial_age에 추가하여 계산할 수 있다.

resident_time = now - response_time;
current_age = corrected_initial_age + resident_time;

4.2.4. Serving Stale Responses

A "stale" response is one that either has explicit expiry information or is allowed to have heuristic expiry calculated, but is not fresh according to the calculations in Section 4.2.

"stale" 응답은 명시적 만료 정보를 가지고 있거나 휴리스틱 만료를 계산할 수 있지만 Section 4.2의 계산에 따라 신선하지 않은 응답이다.

A cache MUST NOT generate a stale response if it is prohibited by an explicit in-protocol directive (e.g., by a "no-store" or “no-cache" cache directive, a "must-revalidate" cache-response-directive, or an applicable "s-maxage" or "proxy-revalidate" cache-response-directive; see Section 5.2.2).

명시적 프로토콜 내 지시어(e.g., "no-store" 또는 "no-cache" 캐시 지시어, "must-revalidate" cache-response-directive 또는 해당 "s-maxage" 또는 "proxy-revalidate" 캐시 cache-response-directive; Section 5.2.2.2 참조)에 의해 금지된 경우 캐시는 오래된 응답을 생성해서는 안 된다.(MUST NOT)

A cache MUST NOT send stale responses unless it is disconnected (i.e., it cannot contact the origin server or otherwise find a forward path) or doing so is explicitly allowed (e.g., by the max-stale request directive; see Section 5.2.1).

캐시의 연결이 끊어진 경우가 아니라면(i.e., 원서버에 접속하거나 다른 경로로 이동할 수 없음) 오래된 응답을 보내서는 안 된다.(MUST NOT) (e.g., max-stale 요청 지시어에 의해, Section 5.2.1 참조).

A cache SHOULD generate a Warning header field with the 110 warn-code (see Section 5.5.1) in stale responses. Likewise, a cache SHOULD generate a 112 warn-code (see Section 5.5.3) in stale responses if the cache is disconnected.

캐시는 오래된 응답으로 110 warn-code(Section 5.5.1 참조)를 사용하여 Warning 헤더를 생성해야 한다.(SHOULD) 마찬가지로 그 캐시의 연결이 끊어진 경우 캐시는 오래된 응답으로 112 warn-code(Section 5.5.3 참조)를 생성해야 한다.(SHOULD)

A cache SHOULD NOT generate a new Warning header field when forwarding a response that does not have an Age header field, even if the response is already stale. A cache need not validate a response that merely became stale in transit.

응답이 이미 오래된 경우에도 Age 헤더 필드가 없는 응답을 전달할 때 캐시가 새 Warning 헤더 필드를 생성하지 않아야 한다.(SHOULD) 캐시는 전송 중에 그저 오래된 응답의 유효성을 확인할 필요가 없다.

4.3. Validation

When a cache has one or more stored responses for a requested URI, but cannot serve any of them (e.g., because they are not fresh, or one cannot be selected; see Section 4.1), it can use the conditional request mechanism [RFC7232] in the forwarded request to give the next inbound server an opportunity to select a valid stored response to use, updating the stored metadata in the process, or to replace the stored response(s) with a new response. This process is known as "validating" or "revalidating" the stored response.

캐시가 요청된 URI에 대해 하나 이상의 저장된 응답을 가지고 있지만 해당 응답을 제공할 수 없는 경우(e.g., 새로운 응답이 아니므로 또는 하나를 선택할 수 없으므로, Section 4.1 참조), 전달된 요청의 조건부 요청 메커니즘 [RFC7232]을 사용하여 다음 인바운드 서버에서 사용위해 유효한 저장된 응답을 선택하거나 절차안에서 저장된 메타데이터를 수정하거나 저장된 응답을 새 응답으로 바꾸기 위한 기회를 제공할 수 있다. 이 절차를 "validating(이하 검증)" 또는 "revalidating(이하 재검토)"이라고 한다.

4.3.1. Sending a Validation Request

When sending a conditional request for cache validation, a cache sends one or more precondition header fields containing validator metadata from its stored

response(s), which is then compared by recipients to determine whether a stored response is equivalent to a current representation of the resource.

캐시 검토를 위한 조건부 요청을 보낼 때 캐시는 저장된 응답으로부터 검증자 메타데이터가 포함된 하나 이상의 전제 조건 헤더 필드를 전송하며, 이 필드를 수신자가 비교하여 저장된 응답이 리소스의 현재 표현과 동일한지 여부를 결정한다.

One such validator is the timestamp given in a Last-Modified header field (Section 2.2 of [RFC7232]), which can be used in an If-Modified-Since header field for response validation, or in an If-Unmodified-Since or If-Range header field for representation selection (i.e., the client is referring specifically to a previously obtained representation with that timestamp).

이러한 검증자 중 하나는 Last-Modified 헤더 필드([RFC7232]의 Section 2.2)에 주어진 타임스탬프로서, 응답 검토에 대한 if-Modified-Since 헤더 필드 또는 표현 선택을 위한 If-Unmodified-Since 또는 If-Range 헤더 필드에서 사용할 수 있다. (i.e., 클라이언트는 그 타임스탬프로 이전에 획득한 표현을 구체적으로 언급하고 있다.)

Another validator is the entity-tag given in an ETag header field (Section 2.3 of [RFC7232]). One or more entity-tags, indicating one or more stored responses, can be used in an If-None-Match header field for response validation, or in an If-Match or If-Range header field for representation selection (i.e., the client is referring specifically to one or more previously obtained representations with the listed entity-tags).

또 다른 검증자는 ETag 헤더 필드에 주어진 entity-tag([RFC7232]의 Section 2.3)이다. 하나 이상의 저장된 응답을 나타내는 하나 이상의 entity-tag를 응답 검토를 위해 If-None-Match 헤더 필드 또는 If-Range 헤더 필드에서 사용할 수 있으며, 표현 선택을 위해 If-Match 또는 If-Range 헤더 필드에서 사용할 수 있다.(i.e., 클라이언트는 나열된 entity-tag와 이전에 획득한 하나 이상의 표현을 특별히 참조하고 있음)

4.3.2. Handling a Received Validation Request

Each client in the request chain may have its own cache, so it is common for a cache at an intermediary to receive conditional requests from other (outbound) caches. Likewise, some user agents make use of conditional requests to limit data transfers

to recently modified representations or to complete the transfer of a partially retrieved representation.

요청 체인의 각 클라이언트는 자체 캐시를 가질 수 있으므로, 중개자의 캐시가 다른 (아웃바운드) 캐시로부터 조건부 요청을 수신하는 것이 일반적이다. 마찬가지로, 일부 사용자 에이전트는 최근에 수정된 표현으로 데이터 전송을 제한하거나 또는 부분적으로 검색된 표현으로 전송을 완료하기 위해 조건부 요청을 사용한다.

If a cache receives a request that can be satisfied by reusing one of its stored 200 (OK) or 206 (Partial Content) responses, the cache SHOULD evaluate any applicable conditional header field preconditions received in that request with respect to the corresponding validators contained within the selected response. A cache MUST NOT evaluate conditional header fields that are only applicable to an origin server, found in a request with semantics that cannot be satisfied with a cached response, or applied to a target resource for which it has no stored responses; such preconditions are likely intended for some other (inbound) server.

캐시가 저장된 200(OK) 또는 206(Partial Content) 응답 중 하나를 재사용하여 충족할 수 있는 요청을 수신하는 경우, 캐시는 선택한 응답에 포함된 해당 검증자에 대해 해당 요청에서 수신된 해당 조건부 헤더 필드 전제조건을 평가해야 한다.(SHOULD) 캐시는 원서버에만 적용 가능한 조건부 헤더 필드를 평가해서는 안 되며, 캐시된 응답으로 충족할 수 없는 의미론 또는 저장된 응답이 없는 대상 리소스에 적용해서는 안 된다. 이러한 전제조건은 다른 (인바운드) 서버를 위한 것일 가능성이 높다.

The proper evaluation of conditional requests by a cache depends on the received precondition header fields and their precedence, as defined in Section 6 of [RFC7232]. The If-Match and If-Unmodified-Since conditional header fields are not applicable to a cache.

캐시에 의한 조건부 요청의 적절한 평가는 [RFC7232]의 Section 6에 정의된 대로 수신된 전제 조건 헤더 필드와 그 우선순위에 따라 달라진다. If-Match 및 If-Unmodified-Since 조건부 헤더 필드는 캐시에 적용되지 않는다.

A request containing an If-None-Match header field (Section 3.2 of [RFC7232]) indicates that the client wants to validate one or more of its own stored responses in comparison to whichever stored response is selected by the cache. If the field-value is "*", or if the field-value is a list of entity-tags and at least one of them matches the entity-tag of the selected stored response, a cache recipient SHOULD generate a 304 (Not Modified) response (using the metadata of the selected stored response) instead of sending that stored response.

If-None-Match 헤더 필드([RFC7232]의 Section 3.2)가 포함된 요청은 클라이언트가 캐시에 의해 선택된 저장된 응답과 비교하여 하나 이상의 자체 저장된 응답을 검증하려고 함을 나타낸다. 필드 값이 "*" 이거나 또는 필드 값이 entity-tag 목록이고 그 중 하나 이상이 선택된 저장된 응답의 entity-tag와 일치하는 경우, 캐시 수신자는 저장된 응답을 보내는 대신 304(Not Modified) 응답(선택한 저장된 응답의 메타데이터 사용)을 생성해야 한다.(SHOULD)

When a cache decides to revalidate its own stored responses for a request that contains an If-None-Match list of entity-tags, the cache MAY combine the received list with a list of entity-tags from its own stored set of responses (fresh or stale) and send the union of the two lists as a replacement If-None-Match header field value in the forwarded request. If a stored response contains only partial content, the cache MUST NOT include its entity-tag in the union unless the request is for a range that would be fully satisfied by that partial stored response. If the response to the forwarded request is 304 (Not Modified) and has an ETag header field value with an entity-tag that is not in the client's list, the cache MUST generate a 200 (OK) response for the client by reusing its corresponding stored response, as updated by the 304 response metadata (Section 4.3.4).

캐시가 entity-tags의 If-None-Match 목록이 포함된 요청에 대해 자체 저장된 응답을 재확정하기로 결정할 때, 캐시는 수신된 목록과 자체 저장된 응답 집합의 entity-tag 목록(신선한 또는 오래된)을 결합하고 전달된 요청에서 두 목록의 결합을 대체 If-None-Match 헤더 필드 값으로 보낼 수 있다. 저장된 응답에 부분적인 내용만 포함된 경우, 부분적인 저장된 응답에 의해 완전히 충족되는 범위에 대한 요청이 없는 한 캐시는 entity-tag를 조합에 포함해서는 안된다. 전달된 요청에 대한 응답이 304(Not Modified)이고 클라이언트 목록에 없는 entity-tag가 있는 ETag 헤더 필드 값이 있는 경우, 캐시는 304 응답 메타데이터(Section 4.3.4)에 의해 업데이트한 대로, 해당 저장된 응답을 재사용하여 클라이언트에 대한 200(OK) 응답을 생성해야 한다.

If an If-None-Match header field is not present, a request containing an If-Modified-Since header field (Section 3.3 of [RFC7232]) indicates that the client wants to validate one or more of its own stored responses by modification date. A cache recipient SHOULD generate a 304 (Not Modified) response (using the metadata of the selected stored response) if one of the following cases is true: 1) the selected stored response has a Last-Modified field-value that is earlier than or equal to the conditional timestamp; 2) no Last-Modified field is present in the selected stored response, but it has a Date field-value that is earlier than or equal to the conditional timestamp; or, 3) neither Last-Modified nor Date is present in the selected stored response, but the cache recorded it as having been received at a time earlier than or equal to the conditional timestamp.

If-None-Match 헤더 필드가 없는 경우, If-Modified-Since 헤더 필드([RFC7232] Section 3.3)를 포함하는 요청은 클라이언트가 수정 날짜까지 하나 이상의 저장된 응답을 검증하기를 원함을 나타낸다.
캐시 수신자는 다음 사례 중 하나가 참인 경우 304(Not Modified) 응답을 생성(선택된 저장된 응답의 메타데이터 사용)해야 한다.(SHOULD)
1) 선택된 저장된 응답에 조건부 타임스탬프보다 이전 또는 같은 Last-Modified field-value가 있음
2) 선택된 저장된 응답에 Last-Modified 필드가 없음, 그러나 조건부 타임스탬프보다 빠르거나 같은 Date field-value을 가지고 있다. 또는
3) 선택된 저장된 응답에 Last-Modified도 없고 Date가 없지만, 캐시는 조건부 타임스탬프보다 동일하거나, 빠른 시간에 수신된 것으로 기록한다.

A cache that implements partial responses to range requests, as defined in [RFC7233], also needs to evaluate a received If-Range header field (Section 3.2 of [RFC7233]) with respect to its selected stored response.

[RFC7233]에서 정의한 범위 요청에 대한 부분 응답을 구현하는 캐시는 또한 선택된 저장된 응답과 관련하여 수신된 If-Range 헤더 필드([RFC7233]의 Section 3.2)를 평가할 필요가 있다.

4.3.3. Handling a Validation Response

Cache handling of a response to a conditional request is dependent upon its status code:

조건부 요청에 대한 응답의 캐시 처리 상태 코드에 따라 다름:

o A 304 (Not Modified) response status code indicates that the stored response can be updated and reused; see Section 4.3.4.

o 304 (Not Modified) 응답 상태 코드는 저장된 응답을 업데이트하고 재사용할 수 있음을 나타낸다. Section 4.3.4를 참조한다.

o A full response (i.e., one with a payload body) indicates that none of the stored responses nominated in the conditional request is suitable. Instead, the cache

MUST use the full response to satisfy the request and MAY replace the stored response(s).

o 완전한 응답(i.e., 페이로드 본문이 있는 응답)은 조건부 요청에서 지명된 저장된 응답 중 어느 것도 적합하지 않음을 나타낸다. 대신, 캐시는 요청을 충족시키기 위해 전체 응답을 사용해야 하며(MUST) 저장된 응답을 대체할 수 있다.(MAY)

o However, if a cache receives a 5xx (Server Error) response while attempting to validate a response, it can either forward this response to the requesting client, or act as if the server failed to respond. In the latter case, the cache MAY send a previously stored response (see Section 4.2.4).

o 그러나 캐시가 응답의 유효성을 확인하려고 시도하는 동안 5xx (Server Error) 응답을 수신하는 경우, 이 응답을 요청 클라이언트에 전달하거나 서버가 응답하지 못한 것처럼 행동할 수 있다. 후자의 경우, 캐시는 이전에 저장된 응답을 보낼 수 있다.(MAY) (Section 4.2.4 참조).

4.3.4. Freshening Stored Responses upon Validation

When a cache receives a 304 (Not Modified) response and already has one or more stored 200 (OK) responses for the same cache key, the cache needs to identify which of the stored responses are updated by this new response and then update the stored response(s) with the new information provided in the 304 response.

캐시가 304(Not Modified) 응답을 수신하고 동일한 캐시 키에 대해 하나 이상의 저장된 200(OK) 응답을 이미 가지고 있는 경우, 캐시는 이 새로운 응답에 의해 갱신되는 저장된 응답 중 어떤 것을 식별한 후 저장된 응답을 304 응답에 제공된 새로운 정보로 갱신해야 한다.

The stored response to update is identified by using the first match (if any) of the following:

업데이트에 대한 저장된 응답은 다음 중 첫 번째 일치(있는 경우)를 사용하여 식별된다.

o If the new response contains a strong validator (see Section 2.1 of [RFC7232]), then that strong validator identifies the selected representation for update. All of the stored responses with the same strong validator are selected. If none of the

stored responses contain the same strong validator, then the cache MUST NOT use the new response to update any stored responses.

o 새 응답에 강한 검증자가 포함된 경우([RFC7232]의 Section 2.1 참조), 해당 강한 검증자가 갱신을 위해 선택된 표현을 식별한다. 동일한 강한 검증자를 사용하여 저장된 모든 응답을 선택한다. 저장된 응답 중 동일한 강한 검증자를 포함하는 응답이 없는 경우, 캐시는 저장된 응답을 갱신하기 위해 새 응답을 사용해서는 안 된다.(MUST NOT)

o If the new response contains a weak validator and that validator corresponds to one of the cache's stored responses, then the most recent of those matching stored responses is selected for update.

o 새 응답에 약한 검증자가 포함되어 있고 해당 검증자가 캐시의 저장된 응답 중 하나에 해당하는 경우 저장된 응답과 일치하는 가장 최근의 응답은 갱신을 위해 선택된다.

o If the new response does not include any form of validator (such as in the case where a client generates an If-Modified-Since request from a source other than the Last-Modified response header field), and there is only one stored response, and that stored response also lacks a validator, then that stored response is selected for update.

o 새 응답에 어떤 형태의 검증자가(예: 클라이언트가 Last-Modified 응답 헤더 필드가 아닌 소스에서 If-Modified-Since 요청을 생성하는 경우) 포함되어 있지 않고 저장된 응답만 하나뿐이고 저장된 응답에도 검증자가 없는 경우 저장된 응답을 선택하여 갱신한다.

If a stored response is selected for update, the cache MUST:

갱신을 위해 저장된 응답을 선택한 경우 캐시는 다음을 수행해야 한다.(MUST)

o delete any Warning header fields in the stored response with warn-code 1xx (see Section 5.5);

o warn-code 1xx를 사용하여 저장된 응답의 모든 Warning 헤더 필드를 삭제한다(Section 5.5 참조).

o retain any Warning header fields in the stored response with warn-code 2xx; and,

o warn-code 2xx를 사용하여 저장된 응답의 모든 Warning 헤더 필드를 유지하며,

o use other header fields provided in the 304 (Not Modified) response to replace all instances of the corresponding header fields in the stored response.

o 304 (Not Modified) 응답에 제공된 다른 헤더 필드를 사용하여 저장된 응답에 있는 해당 헤더 필드의 모든 인스턴스를 교체한다.

4.3.5. Freshening Responses via HEAD

A response to the HEAD method is identical to what an equivalent request made with a GET would have been, except it lacks a body. This property of HEAD responses can be used to invalidate or update a cached GET response if the more efficient conditional GET request mechanism is not available (due to no validators being present in the stored response) or if transmission of the representation body is not desired even if it has changed.

HEAD 메서드에 대한 응답은 GET와 동등한 요청이 있었을 것과 동일하지만, 단, 본문이 없다는 것은 제외한다. 이 HEAD 응답의 속성은 보다 효율적인 조건부 GET 요청 메커니즘을 사용할 수 없거나(저장된 응답에 검증자가 없기 때문에) 변경되었더라도 표현 본문의 전송을 원하지 않는 경우 캐시된 GET 응답을 무효화하거나 갱신하는 데 사용할 수 있다.

When a cache makes an inbound HEAD request for a given request target and receives a 200 (OK) response, the cache SHOULD update or invalidate each of its stored GET responses that could have been selected for that request (see Section 4.1).

캐시가 특정 요청 대상에 대한 인바운드 HEAD 요청을 하고 200 (OK) 응답을 수신할 때, 캐시는 해당 요청에 대해 선택되었을 수 있는 저장된 GET 응답을 각각 갱신하거나 무효화해야 한다.(SHOULD) (Section 4.1 참조)

For each of the stored responses that could have been selected, if the stored response and HEAD response have matching values for any received validator fields (ETag and Last-Modified) and, if the HEAD response has a Content-Length header field, the value of Content-Length matches that of the stored response, the cache

SHOULD update the stored response as described below; otherwise, the cache SHOULD consider the stored response to be stale.

선택될 수 있는 각 저장된 응답에 대해, 저장된 응답과 HEAD 응답에 수신된 검증자 필드(ETag 및 Last-Modified)와 일치하는 값이 있고, HEAD 응답에 Content-Length 헤더 필드가 있는 경우, 저장된 응답의 값과 일치하는 Content-Length 값이 있으면, 아래에 설명된 대로, 캐시에서 저장된 응답을 수정해야 하며,(SHOULD) 그렇지 않은 경우, 캐시는 저장된 응답이 오래된 것으로 간주해야 한다.(SHOULD)

If a cache updates a stored response with the metadata provided in a HEAD response, the cache MUST:

캐시가 저장된 응답을 HEAD 응답에서 제공된 메타데이터와 같이 갱신하는 경우, 캐시는 다음을 수행해야 한다.(MUST)

o delete any Warning header fields in the stored response with warn-code 1xx (see Section 5.5);

o warn-code 1xx를 사용하여 저장된 응답의 모든 Warning 헤더 필드를 삭제한다 (Section 5.5 참조).

o retain any Warning header fields in the stored response with warn-code 2xx; and,

o warn-code 2xx를 사용하여 저장된 응답의 모든 Warning 헤더 필드를 유지하며,

o use other header fields provided in the HEAD response to replace all instances of the corresponding header fields in the stored response and append new header fields to the stored response's header section unless otherwise restricted by the Cache-Control header field.

o Cache-Control 헤더 필드에 의해 달리 제한되지 않는 한, HEAD 응답에 제공된 다른 헤더 필드를 사용하여 저장된 응답의 헤더 부문에 해당 헤더 필드의 모든 인스턴스를 교체하고 새 헤더 필드를 추가한다.

4.4. Invalidation

Because unsafe request methods (Section 4.2.1 of [RFC7231]) such as PUT, POST or DELETE have the potential for changing state on the origin server, intervening caches can use them to keep their contents up to date.

PUT, POST 또는 DELETE와 같은 안전하지 않은 요청 메서드([RFC7231]의 Section 4.2.1)은 원서버에서 상태를 변경할 가능성이 있기 때문에, 그 사이의 캐시는 내용을 최신 상태로 유지하기 위해 그것들을 사용할 수 있다.

A cache MUST invalidate the effective Request URI (Section 5.5 of [RFC7230]) as well as the URI(s) in the Location and Content-Location response header fields (if present) when a non-error status code is received in response to an unsafe request method.

안전하지 않은 요청 메서드에 대한 응답으로 비-오류 상태 코드가 수신될 때, 캐시는 유효한 요청 URI([RFC7230]의 Section 5.5)와 Location 및 Content-Location 헤더 필드(있는 경우)의 URI를 무효화해야 한다.(MUST)

However, a cache MUST NOT invalidate a URI from a Location or Content-Location response header field if the host part of that URI differs from the host part in the effective request URI (Section 5.5 of [RFC7230]). This helps prevent denial-of-service attacks.

그러나 URI의 호스트 부분이 유효한 요청 URI의 호스트 부분과 다를 경우 Location 또는 Content-Location 응답 헤더 필드에서 URI를 무효화해서는 안 된다.(MUST NOT) ([RFC7230]의 Section 5.5). 이는 denial-of-service 공격을 방지하는 데 도움이 된다.

A cache MUST invalidate the effective request URI (Section 5.5 of [RFC7230]) when it receives a non-error response to a request with a method whose safety is unknown.

캐시는 안전한지 알 수 없는 메서드로 요청에 대한 비-오류 응답을 수신 했을 때 유효한 요청 URI([RFC7230]의 Section 5.5)를 무효화해야 한다.(MUST)

Here, a "non-error response" is one with a 2xx (Successful) or 3xx (Redirection) status code. "Invalidate" means that the cache will either remove all stored responses related to the effective request URI or will mark these as "invalid" and in need of a mandatory validation before they can be sent in response to a subsequent request.

여기서 "none-error response(비-오류 응답)"은 2xx (Successful) 또는 3xx (Redirection) 상태 코드가 있는 응답이다. "invalidate"는 캐시가 유효한 요청 URI와 관련된 저장된 응답을 모두 제거하거나 "invalid"로 표시하고 후속 요청에 대한 응답으로 전송하기 전에 필수 검토가 필요하다는 것을 의미한다.

Note that this does not guarantee that all appropriate responses are invalidated. For example, a state-changing request might invalidate responses in the caches it travels through, but relevant responses still might be stored in other caches that it has not.

그렇다고 모든 적절한 응답이 무효화되는 것은 아니라는 점에 유의한다. 예를 들어, 상태 변경 요청은 그것이 여행하는 캐시의 응답을 무효화할 수 있지만, 관련 응답은 여전히 그렇지 않은 다른 캐시에 저장될 수 있다.

5. Header Field Definitions

This section defines the syntax and semantics of HTTP/1.1 header fields related to caching.

이 절에서는 캐싱과 관련된 HTTP/1.1 헤더 필드의 구문과 의미론을 정의한다.

5.1. Age

The "Age" header field conveys the sender's estimate of the amount of time since the response was generated or successfully validated at the origin server. Age values are calculated as specified in Section 4.2.3.

"Age" 헤더 필드는 발신자로부터 응답이 생성되거나 원서버에서 성공적으로 검증된 이후 시간의 추정치를 전달한다. Age 값은 Section 4.2.3에 명시된 대로 계산된다.

Age = delta-seconds

The Age field-value is a non-negative integer, representing time in seconds (see Section 1.2.1).

Age 필드 값은 음수가 아닌 정수로서 시간(초)을 나타낸다(Section 1.2.1 참조).

The presence of an Age header field implies that the response was not generated or validated by the origin server for this request. However, lack of an Age header field does not imply the origin was contacted, since the response might have been received from an HTTP/1.0 cache that does not implement Age.

Age 헤더 필드가 존재한다는 것은 이 요청에 대해 원서버가 응답을 생성하거나 검증하지 않았음을 의미한다. 그러나 Age 헤더 필드가 부족하다고 해서 Age가 구현되지 않는 HTTP/1.0 캐시에서 응답이 수신되었을 수 있기 때문에 원본이 대면된 것은 아니다.

5.2. Cache-Control

The "Cache-Control" header field is used to specify directives for caches along the request/response chain. Such cache directives are unidirectional in that the presence of a directive in a request does not imply that the same directive is to be given in the response.

"Cache-Control" 헤더 필드는 요청/응답 체인을 따라 캐시에 대한 지시어를 지정하는 데 사용된다. 그러한 캐시 지시어는 요청에 지시어가 존재한다고 해서 응답에 동일한 지시어가 주어지는 것은 아니라는 점에서 단방향적이다.

A cache MUST obey the requirements of the Cache-Control directives defined in this section. See Section 5.2.3 for information about how Cache-Control directives defined elsewhere are handled.

캐시는 반드시 이 절에 정의된 캐시 제어 지시어의 요구 사항을 준수해야 한다.(MUST) 다른 곳에서 정의한 Cache-Control 지시어를 처리하는 방법에 대한 자세한 내용은 Section 5.2.3을 참조한다.

Note: Some HTTP/1.0 caches might not implement Cache-Control.

참고: 일부 HTTP/1.0 캐시는 Cache-Control이 구현되어 있지 않을수 있다.

A proxy, whether or not it implements a cache, MUST pass cache directives through in forwarded messages, regardless of their significance to that application, since the directives might be applicable to all recipients along the request/response chain. It is not possible to target a directive to a specific cache.

캐시를 구현하든 구현하지 않든 프락시는 요청/응답 체인을 따라 모든 수신자에게 해당 지시어가 적용될 수 있으므로 해당 애플리케이션에 대한 중요성에 관계없이, 전달된 메시지를 통해 캐시 지시어를 전달해야 한다. 특정 캐시에 대한 지시어를 대상으로 하는 것은 불가능하다.

Cache directives are identified by a token, to be compared case-insensitively, and have an optional argument, that can use both token and quoted-string syntax. For the directives defined below that define arguments, recipients ought to accept both forms, even if one is documented to be preferred. For any directive not defined by this specification, a recipient MUST accept both forms.

캐시 지시어는 토큰으로 식별되며, 대소문자를 구분하지 않고 비교되며 토큰과 quoted-string 구문을 모두 사용할 수 있는 선택적 인수를 갖는다. 인수를 정의하는 아래에 정의된 지시어의 경우, 수신자는 선호되는 것으로 문서화되더라도 두 가지 양식을 모두 받아들여야 한다. 이 명세에 의해 정의되지 않은 모든 지시어에 대해 수신자는 반드시 두 가지 양식을 모두 수용해야 한다.(MUST)

```
Cache-Control   = 1#cache-directive

cache-directive = token [ "=" ( token / quoted-string ) ]
```

For the cache directives defined below, no argument is defined (nor allowed) unless stated otherwise.

아래에 정의된 캐시 지시어에 대해서는 달리 명시되지 않는 한 인수가 정의되지 않는다(또는 허용되지 않는다).

5.2.1. Request Cache-Control Directives

5.2.1.1. max-age

Argument syntax:

delta-seconds (see Section 1.2.1)

The "max-age" request directive indicates that the client is unwilling to accept a response whose age is greater than the specified number of seconds. Unless the max-stale request directive is also present, the client is not willing to accept a stale response.

"max-age" 요청 지시어는 클라이언트가 지정된 시간(초)보다 나이가 많은 응답을 받아들일 의사가 없음을 나타낸다. max-stale 요청 지침도 존재하지 않는 한, 클라이언트는 오래된 응답을 받아들이려 하지 않는다.

This directive uses the token form of the argument syntax: e.g., 'max-age=5' not 'max-age="5"'. A sender SHOULD NOT generate the quoted-string form.

이 지시어는 인수 구문의 토큰 형태(예: 'max-age="5"'가 아닌 'max-age=5')를 사용한다. 발신자는 quoted-string 양식을 생성해서는 안 된다.(SHOULD NOT)

5.2.1.2. max-stale

Argument syntax:

delta-seconds (see Section 1.2.1)

The "max-stale" request directive indicates that the client is willing to accept a response that has exceeded its freshness lifetime. If max-stale is assigned a value, then the client is willing to accept a response that has exceeded its freshness lifetime by no more than the specified number of seconds. If no value is assigned to max-stale, then the client is willing to accept a stale response of any age.

"max-stale" 요청 지시어는 클라이언트의 신선도 수명을 초과한 응답을 받아들일 용의가 있음을 나타낸다. max-stale이 값을 할당받으면 클라이언트는 지정된 초 수명을 초과하지 않는 응답을 기꺼이 수락할 것이다. max-stale에 값이 할당되지 않으면 클라이언트는 어떤 나이의 오래된 응답을 기꺼이 수락할 것이다.

This directive uses the token form of the argument syntax: e.g., 'max-stale=10' not 'max-stale="10"'. A sender SHOULD NOT generate the quoted-string form.

이 지시어는 인수 구문의 토큰 형태(예: 'max-stale="10"'이 아닌 'max-stale=10'을 사용한다. 발신자는 quoted-string 양식을 생성해서는 안 된다.(SHOULD NOT)

5.2.1.3. min-fresh

Argument syntax:

delta-seconds (see Section 1.2.1)

The "min-fresh" request directive indicates that the client is willing to accept a response whose freshness lifetime is no less than its current age plus the specified time in seconds. That is, the client wants a response that will still be fresh for at least the specified number of seconds.

"min-fresh" 요청 지시어는 클라이언트가 신선도 수명이 현재 나이와 지정된 시간(초) 이하인 응답을 기꺼이 수락할 의사가 있음을 나타낸다. 즉, 클라이언트는 적어도 지정된 시간(초) 동안 신선할 응답을 원한다.

This directive uses the token form of the argument syntax: e.g., 'min-fresh=20' not 'min-fresh="20"'. A sender SHOULD NOT generate the quoted-string form.

이 지시어는 인수 구문의 토큰 형식을 사용한다. 예를 들어 'min-fresh="20"'이 아니라 'min-fresh='20'이다. 발신자는 quoted-string 양식을 생성해서는 안 된다.(SHOULD NOT)

5.2.1.4. no-cache

The "no-cache" request directive indicates that a cache MUST NOT use a stored response to satisfy the request without successful validation on the origin server.

"no-cache" 요청 지시어는 캐시가 원서버에 대한 성공적인 검토 없이 요청을 충족하기 위해 저장된 응답을 사용해서는 안 된다(MUST NOT)는 것을 나타낸다.

5.2.1.5. no-store

The "no-store" request directive indicates that a cache MUST NOT store any part of either this request or any response to it. This directive applies to both private and shared caches. "MUST NOT store" in this context means that the cache MUST NOT intentionally store the information in non-volatile storage, and MUST make a best-effort attempt to remove the information from volatile storage as promptly as possible after forwarding it.

"no-store" 요청 지시어는 캐시가 이 요청의 일부 또는 그에 대한 응답을 저장해서는 안 된다(MUST NOT)는 것을 나타낸다. 이 지시어는 개인 캐시와 공유 캐시에 모두 적용된다. 이 맥락에서 "저장해서는 안 된다(MUST NOT)"는 것은 캐시가 의도적으로 정보를 비휘발성 저장소에 저장해서는 안 된다(MUST NOT)는 것을 의미하며, 정보를 전달 후 가능한 한 신속하게 휘발성 저장소에서 정보를 제거하기 위해 최선의 시도를 해야 한다.(MUST)

This directive is NOT a reliable or sufficient mechanism for ensuring privacy. In particular, malicious or compromised caches might not recognize or obey this directive, and communications networks might be vulnerable to eavesdropping.

이 지시어는 사생활을 보장하기 위한 신뢰할 수 있거나 충분한 메커니즘이 아니다. 특히, 악의적이거나 손상된 캐시는 이 지시어를 인식하거나 따르지 않을 수 있으며, 통신망은 도청에 취약할 수 있다.

Note that if a request containing this directive is satisfied from a cache, the no-store request directive does not apply to the already stored response.

이 지시어를 포함하는 요청이 캐시에서 충족되는 경우, 이미 저장된 응답에는 no-store 요청 지시어가 적용되지 않는다는 점에 참고한다.

5.2.1.6. no-transform

The "no-transform" request directive indicates that an intermediary (whether or not it implements a cache) MUST NOT transform the payload, as defined in Section 5.7.2 of [RFC7230].

"no-transform" 요청 지시어는 [RFC7230]의 Section 5.7.2에 정의된 대로 중간자(캐시를 구현하는지 여부)가 페이로드의 변환을 해서는 안 된다(MUST NOT)는 것을 나타낸다.

5.2.1.7. only-if-cached

The "only-if-cached" request directive indicates that the client only wishes to obtain a stored response. If it receives this directive, a cache SHOULD either respond using a stored response that is consistent with the other constraints of the request, or respond with a 504 (Gateway Timeout) status code. If a group of caches is being operated as a unified system with good internal connectivity, a member cache MAY forward such a request within that group of caches.

"only-if-cached" 요청 지시어는 클라이언트가 저장된 응답만 얻기를 원한다는 것을 나타낸다. 이 지시어를 수신할 경우 캐시는 요청의 다른 제약조건과 일치하는 저장된 응답을 사용하여 응답하거나 504 (Gateway Timeout) 상태 코드로 응답해야 한다. 캐시 그룹이 내부 커넥션이 양호한 통합 시스템으로 운영되는 경우, 멤버 캐시는 해당 캐시 그룹 내에서 이러한 요청을 전달할 수 있을 것이다.(MAY)

5.2.2. Response Cache-Control Directives

5.2.2.1. must-revalidate

The "must-revalidate" response directive indicates that once it has become stale, a cache MUST NOT use the response to satisfy subsequent requests without successful validation on the origin server.

"must-revalidate" 응답 지시어는 일단 오래되면 캐시가 원서버에 대한 성공적인 검토 없이 후속 요청을 충족하기 위해 응답을 사용해서는 안 된다(MUST NOT)는 것을 나타낸다.

The must-revalidate directive is necessary to support reliable operation for certain protocol features. In all circumstances a cache MUST obey the must-revalidate directive; in particular, if a cache cannot reach the origin server for any reason, it MUST generate a 504 (Gateway Timeout) response.

must-revalidate 지시어는 특정 프로토콜 기능에 대한 신뢰성 있는 작동을 지원하기 위해 필요하다. 모든 상황에서 캐시는 must-revalidate 지시어를 반드시 준수해야 한다.(MUST) 특히 어떤 이유로든 캐시가 원서버에 도달할 수 없는 경우, 반드시 504 (Gateway Timeout) 응답을 생성해야 한다.(MUST)

The must-revalidate directive ought to be used by servers if and only if failure to validate a request on the representation could result in incorrect operation, such as a silently unexecuted financial transaction.

표현에 대한 요청의 유효성을 검증하지 못할 경우에, 그리고 조용히 실행되지 않은 경제적 트랜잭션과 같이 잘못된 운영으로 이어질 수 있을 때만, must-revalidate 지시어를 서버에서 사용해야 한다.

5.2.2.2. no-cache

Argument syntax:

#field-name

The "no-cache" response directive indicates that the response MUST NOT be used to satisfy a subsequent request without successful validation on the origin server. This allows an origin server to prevent a cache from using it to satisfy a request without contacting it, even by caches that have been configured to send stale responses.

"no-cache" 응답 지시어는 원서버에 대한 성공적인 검토 없이 후속 요청을 충족하기 위해 응답을 사용해서는 안 된다는 것을 나타낸다.(MUST NOT) 이렇게 하면 오래된 응답을 보내도록 구성된 캐시에 의해서도 캐시가 접속하지 않고 요청을 충족시키기 위해 캐시를 사용하는 것을 원서버가 방지할 수 있다.

If the no-cache response directive specifies one or more field-names, then a cache MAY use the response to satisfy a subsequent request, subject to any other restrictions on caching. However, any header fields in the response that have the field-name(s) listed MUST NOT be sent in the response to a subsequent request without successful revalidation with the origin server. This allows an origin server to prevent the re-use of certain header fields in a response, while still allowing caching of the rest of the response.

no-cache 응답 지시어가 하나 이상의 필드 이름을 지정하는 경우 캐시에 대한 다른 제한에 따라 캐시가 응답을 사용하여 후속 요청을 충족할 수 있다. 그러나 필드 이름이 나열된 응답의 헤더 필드는 원서버와의 성공적인 재검토를 거치지 않고 후속 요청에 대한 응답으로 전송되어서는 안 된다.(MUST NOT) 이렇게 하면 원서버는 응답에서 특정 헤더 필드의 재사용을 방지하는 동시에 나머지 응답의 캐싱을 허용한다.

The field-names given are not limited to the set of header fields defined by this specification. Field names are case-insensitive.

주어진 필드 이름은 이 명세에 의해 정의된 헤더 필드 집합에만 국한되지 않는다. 필드 이름은 대소문자를 구분하지 않는다.

This directive uses the quoted-string form of the argument syntax. A sender SHOULD NOT generate the token form (even if quoting appears not to be needed for single-entry lists).

이 지시어는 quoted-string 양식의 인수 구문을 사용한다. 발신자는 토큰 양식을 생성해서는 안 된다. (SHOULD NOT) (single-entry 목록에 인용이 필요하지 않은 경우에도)

Note: Although it has been back-ported to many implementations, some HTTP/1.0 caches will not recognize or obey this directive. Also, no-cache response directives with field-names are often handled by caches as if an unqualified no-cache directive was received; i.e., the special handling for the qualified form is not widely implemented.

참고: 많은 구현에 역-포팅되었지만, 일부 HTTP/1.0 캐시는 이 지시어를 인식하거나 준수하지 않는다. 또한 field-name을 가진 no-cache 지시어는 자격없는 no-cache 지시어가 수신된 것처럼 캐시에 의해 처리되는 경우가 많다. 즉, 적격 양식에 대한 특별 취급은 널리 시행되지 않는다.

5.2.2.3. no-store

The "no-store" response directive indicates that a cache MUST NOT store any part of either the immediate request or response. This directive applies to both private and shared caches. "MUST NOT store" in this context means that the cache MUST NOT intentionally store the information in non-volatile storage, and MUST make a best-effort attempt to remove the information from volatile storage as promptly as possible after forwarding it.

"no-store" 응답 지시어는 캐시가 즉각적인 요청 또는 응답의 일부를 저장해서는 안 된다(MUST NOT)는 것을 나타낸다. 이 지시어는 개인 캐시와 공유 캐시에 모두 적용된다. 이 맥락에서 "저장해서는 안 된다(MUST NOT)"는 것은 캐시가 의도적으로 정보를 비휘발성 저장소에 저장해서는 안 된다(MUST NOT)는 것을 의미하며, 정보를 전달 후 가능한 한 신속하게 휘발성 저장소에서 정보를 제거하기 위해 최선의 시도를 해야 한다.(MUST)

This directive is NOT a reliable or sufficient mechanism for ensuring privacy. In particular, malicious or compromised caches might not recognize or obey this directive, and communications networks might be vulnerable to eavesdropping.

이 지시어는 사생활을 보장하기 위한 신뢰할 수 있거나 충분한 메커니즘이 아니다. 특히, 악의적이거나 손상된 캐시는 이 지시어를 인식하거나 따르지 않을 수 있으며, 통신망은 도청에 취약할 수 있다.

5.2.2.4. no-transform

The "no-transform" response directive indicates that an intermediary (regardless of whether it implements a cache) MUST NOT transform the payload, as defined in Section 5.7.2 of [RFC7230].

"no-transform" 응답 지시어는 [RFC7230]의 Section 5.7.2에서 정의한 대로 (캐시를 구현하는지 여부에 관계 없이) 중개자가 페이로드의 변형을 해서는 안 된다는 것을 나타낸다.

5.2.2.5. public

The "public" response directive indicates that any cache MAY store the response, even if the response would normally be non-cacheable or cacheable only within a private cache. (See Section 3.2 for additional details related to the use of public in response to a request containing Authorization, and Section 3 for details of how public affects responses that would normally not be stored, due to their status codes not being defined as cacheable by default; see Section 4.2.2.)

"public" 응답 지시어는 일반적으로 응답이 캐시 불가능하거나 개인 캐시 내에서만 캐시 가능하더라도 어떤 캐시도 응답을 저장할 수 있음을 나타낸다. (Authorization을 포함한 요청에 대한 범용적 사용과 관련된 추가 세부 정보는 Section 3.2를 참조하고, 상태 코드가 기본적으로 캐시 가능으로 정의되지 않기 때문에,(Section 4.2.2 참조) 일반적으로 저장되지 않는 응답에 대한 공개가 영향을 미치는 방법에 대한 세부 정보는 Section 3을 참조한다.)

5.2.2.6. private

Argument syntax:

```
#field-name
```

The "private" response directive indicates that the response message is intended for a single user and MUST NOT be stored by a shared cache. A private cache MAY store the response and reuse it for later requests, even if the response would normally be non-cacheable.

"private" 응답 지시어는 응답 메시지가 단일 사용자를 위한 것이며 공유 캐시에 저장되어서는 안 된다(MUST NOT)는 것을 나타낸다. 개인 캐시 응답을 저장하고, 일반적으로 캐시할 수 없는 응답이라도 이후 요청에 재사용할 것 이다.(MAY)

If the private response directive specifies one or more field-names, this requirement is limited to the field-values associated with the listed response header fields. That is, a shared cache MUST NOT store the specified field-names(s), whereas it MAY store the remainder of the response message.

ptivate 응답 지시어가 하나 이상의 field-names을 지정하는 경우, 이 요건은 나열된 응답 헤더 필드와 관련된 field-value로 제한된다. 즉, 공유 캐시는 지정된 field-name(s)을 저장해서는 안 되나,(MUST NOT) 반면에 응답 메시지의 나머지 부분은 저장할 것이다.(MAY)

The field-names given are not limited to the set of header fields defined by this specification. Field names are case-insensitive.

주어진 field-names은 이 명세에 의해 정의된 헤더 필드 집합에만 국한되지 않는다. 필드 이름은 대소문자를 구분하지 않는다.

This directive uses the quoted-string form of the argument syntax. A sender SHOULD NOT generate the token form (even if quoting appears not to be needed for single-entry lists).

이 지시어는 quoted-string 양식의 인수 구문을 사용한다. 발신자는 토큰 양식을 생성해서는 안 된다. (SHOULD NOT) (single-entry 목록에 인용이 필요하지 않은 경우에도)

Note: This usage of the word "private" only controls where the response can be stored; it cannot ensure the privacy of the message content. Also, private response directives with field-names are often handled by caches as if an unqualified private directive was received; i.e., the special handling for the qualified form is not widely implemented.

참고: "private"이라는 단어의 사용은 응답을 어디에 저장할 수 있는지만 제어하며, 메시지 내용의 개인 정보를 보장할 수 없다. 또한 field-names을 가진 private 응답 지시어는 자격이

없는 private 지시어가 수신된 것처럼 캐시에 의해 처리되는 경우가 많다. 즉, 자격 있는 형식에 대한 특별 취급은 널리 시행되지 않는다.

5.2.2.7. proxy-revalidate

The "proxy-revalidate" response directive has the same meaning as the must-revalidate response directive, except that it does not apply to private caches.

"proxy-revalidate" 응답 지시어는 개인 캐시에 적용되지 않는다는 점을 제외하고 must-revalidate 응답 지시어와 동일한 의미를 갖는다.

5.2.2.8. max-age

Argument syntax:

delta-seconds (see Section 1.2.1)

The "max-age" response directive indicates that the response is to be considered stale after its age is greater than the specified number of seconds.

"max-age" 응답 지시어는 해당 나이가 지정된 초보다 큰 이후의 응답이 오래된 것으로 간주되어야 함을 나타낸다.

This directive uses the token form of the argument syntax: e.g., 'max-age=5' not 'max-age="5"'. A sender SHOULD NOT generate the quoted-string form.

이 지시어는 인수 구문의 토큰 형태(예: 'max-age="5"'가 아닌 'max-age=5')를 사용한다. 발신자는 quoted-string 양식을 생성해서는 안 된다.(SHOULD NOT)

5.2.2.9. s-maxage

Argument syntax:

delta-seconds (see Section 1.2.1)

The "s-maxage" response directive indicates that, in shared caches, the maximum age specified by this directive overrides the maximum age specified by either the max-age directive or the Expires header field. The s-maxage directive also implies the semantics of the proxy-revalidate response directive.

"s-maxage" 응답 지시어는 공유 캐시에서 이 지시어로 지정한 최대 나이가 max-age 지시어 또는 Expires 헤더 필드에서 지정한 최대 나이를 재정의함을 나타낸다. s-maxage 지시어 또한 proxy-revalidate 응답 지시어의 의미론도 내포하고 있다.

This directive uses the token form of the argument syntax: e.g., 's-maxage=10' not 's-maxage="10"'. A sender SHOULD NOT generate the quoted-string form.

이 지시어는 인수 구문의 토큰 형태(예: 's-maxage="10"'이 아닌 's-maxage=10')를 사용한다. 발신자는 quoted-string 양식을 생성해서는 안 된다.

5.2.3. Cache Control Extensions

The Cache-Control header field can be extended through the use of one or more cache-extension tokens, each with an optional value. A cache MUST ignore unrecognized cache directives.

Cache-Control 헤더 필드는 각각 옵션 값을 가진 하나 이상의 캐시 확장 토큰을 사용하여 확장할 수 있다. 캐시는 인식되지 않는 캐시 지시어를 무시해야 한다.(MUST)

Informational extensions (those that do not require a change in cache behavior) can be added without changing the semantics of other directives.

정보 확장(캐시 동작의 변경이 필요하지 않은 확장)은 다른 지시어의 의미론 변경 없이 추가할 수 있다.

Behavioral extensions are designed to work by acting as modifiers to the existing base of cache directives. Both the new directive and the old directive are supplied, such that applications that do not understand the new directive will default to the behavior specified by the old directive, and those that understand the new directive will recognize it as modifying the requirements associated with the old directive. In this way, extensions to the existing cache-control directives can be made without breaking deployed caches.

행동 확장은 기존 캐시 지시어의 기본에 대한 수정자 역할을 함으로써 작동하도록 설계된다. 새로운 지시어와 구 지시어 모두 제공되며, 새로운 지시어를 이해하지 못하는 애플리케이션은 구 지시어에 의해 지정된 행동을 기본값으로 할 것이며, 새로운 지시어를 이해하는 애플리케이션은 구 지시어와 관련된 요건을 수정하는 것으로 인식할 것이다. 이렇게 하면 배치된 캐시를 깨지 않고도 기존 캐시 제어 명령으로 확장할 수 있다.

For example, consider a hypothetical new response directive called "community" that acts as a modifier to the private directive: in addition to private caches, any cache that is shared only by members of the named community is allowed to cache the response. An origin server wishing to allow the UCI community to use an otherwise private response in their shared cache(s) could do so by including

예를 들어, private 지시어의 수식어 역할을 하는 가상의 새로운 응답 지시어 "community"를 생각해 보자. 개인 캐시 외에, 명명된 커뮤니티의 구성원만이 공유하는 모든 캐시는 응답을 캐시할 수 있다. UCI 커뮤니티가 공유 캐시에서 다른 개인 응답을 사용할 수 있도록 허용하려는 원서버

```
Cache-Control: private, community="UCI"
```

A cache that recognizes such a community cache-extension could broaden its behavior in accordance with that extension. A cache that does not recognize the community cache-extension would ignore it and adhere to the private directive.

그러한 커뮤니티 캐시 확장을 인식하는 캐시는 그 확장에 따라 그것의 행동을 넓힐 수 있다. 커뮤니티 캐시 확장을 인식하지 못하는 캐시는 이를 무시하고 private 지시어를 준수할 것이다.

5.3. Expires

The "Expires" header field gives the date/time after which the response is considered stale. See Section 4.2 for further discussion of the freshness model.

"Expires" 헤더 필드는 응답이 오래된 것으로 간주되는 날짜/시간을 제공한다. 신선도 모델에 대한 자세한 설명은 Section 4.2를 참조한다.

The presence of an Expires field does not imply that the original resource will change or cease to exist at, before, or after that time.

Expires 필드가 있다고 해서 원래 리소스가 변경되거나 그 시간 이전 또는 이후에 존재한다는 의미는 아니다.

The Expires value is an HTTP-date timestamp, as defined in Section 7.1.1.1 of [RFC7231].

Expires 값은 [RFC7231]의 Section 7.1.1에서 정의한 HTTP-date 타임스탬프다.

```
Expires = HTTP-date
```

For example

예를 들어

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

A cache recipient MUST interpret invalid date formats, especially the value "0", as representing a time in the past (i.e., “already expired").

캐시 수신자는 잘못된 날짜 형식, 특히 "0" 값을 과거의 시간(즉, "이미 만료됨")을 나타내는 것으로 해석해야 한다.(MUST)

If a response includes a Cache-Control field with the max-age directive (Section 5.2.2.8), a recipient MUST ignore the Expires field. Likewise, if a response includes the s-maxage directive (Section 5.2.2.9), a shared cache recipient MUST ignore the Expires field. In both these cases, the value in Expires is only intended for recipients that have not yet implemented the Cache-Control field.

응답에 max-age 지시어(Section 5.2.2.8)가 있는 Cache-Control 필드가 포함된 경우, 수신자는 Expires 필드를 무시해야 한다.(MUST) 마찬가지로 응답에 s-maxage 지시어(Section 5.2.2.9)가 포함된 경우 공유 캐시 수신자는 Expires 필드를 무시해야 한다.(MUST) 이 두 가지 경우 모두 Expires 값은 Cache-Control 필드를 아직 구현하지 않은 수신자만을 대상으로 한다.

An origin server without a clock MUST NOT generate an Expires field unless its value represents a fixed time in the past (always expired) or its value has been associated with the resource by a system or user with a reliable clock.

시계가 없는 원서버는 해당 값이 과거(항상 만료)의 고정 시간을 나타내거나 신뢰할 수 있는 시계를 가진 시스템 또는 사용자에 의해 리소스와 연결된 경우가 아니면 Expires 필드를 생성해서는 안 된다.(MUST NOT)

Historically, HTTP required the Expires field-value to be no more than a year in the future. While longer freshness lifetimes are no longer prohibited, extremely large values have been demonstrated to cause problems (e.g., clock overflows due to use of 32-bit integers for time values), and many caches will evict a response far sooner than that.

역사적으로, HTTP는 Expires field-value가 향후 1년 이하가 되도록 요구하였다. 더 긴 신선도 수명은 더 이상 금지되지 않지만, 극도로 큰 값은 문제를 일으키는 것으로 입증되었고 (e.g., 시간 값에 32비트 정수 사용으로 인한 시계 오버플로), 대부분의 캐시가 그보다 훨씬 빠르게 응답을 제거할 것이다.

5.4. Pragma

The "Pragma" header field allows backwards compatibility with HTTP/1.0 caches, so that clients can specify a "no-cache" request that they will understand (as Cache-Control was not defined until HTTP/1.1). When the Cache-Control header field is also present and understood in a request, Pragma is ignored.

"Pragma" 헤더 필드는 HTTP/1.0 캐시와의 역호환성을 허용하므로 클라이언트가 이해할 "no-cache" 요청을 지정할 수 있다(HTTP/1.1까지 Cache-Control이 정의되지 않았기 때문에). Cache-Control 헤더 필드도 존재하며 요청으로 이해되면, Pragma는 무시된다.

In HTTP/1.0, Pragma was defined as an extensible field for implementation-specified directives for recipients. This specification deprecates such extensions to improve interoperability.

HTTP/1.0에서 Pragma는 수신자를 위한 구현 지정 지시어를 위한 확장 가능한 필드로 정의되었다. 이 명세는 상호운용성을 개선하기 위해 그러한 확장을 반대한다.

```
Pragma           = 1#pragma-directive
pragma-directive = "no-cache" / extension-pragma
extension-pragma = token [ "=" ( token / quoted-string ) ]
```

When the Cache-Control header field is not present in a request, caches MUST consider the no-cache request pragma-directive as having the same effect as if "Cache-Control: no-cache" were present (see Section 5.2.1).

캐시는 요청에 Cache-Control 헤더 필드가 없을 때, "Cache-Control: no-cache"이 존재하는 것과 동일한 효과를 갖는 것으로, 캐시는 no-cache 요청을 pragma-directive로 간주해야 한다(Secntion 5.2.1 참조).

When sending a no-cache request, a client ought to include both the pragma and cache-control directives, unless Cache-Control: no-cache is purposefully omitted to target other Cache-Control response directives at HTTP/1.1 caches. For example:

no-cache 요청을 보낼 때, 클라이언트는 Cache-Control : no-cache가 HTTP/1.1 캐시의 다른 Cache-Control 응답 지시어를 의도적으로 생략하지 않는 한, pragma 및 cache-control 지시어를 둘 다 포함해야 한다. 예를 들어:

```
GET / HTTP/1.1
Host: www.example.com
Cache-Control: max-age=30
Pragma: no-cache
```

will constrain HTTP/1.1 caches to serve a response no older than 30 seconds, while precluding implementations that do not understand Cache-Control from serving a cached response.

Cache-Control을 이해하지 못하는 구현이, 캐시된 응답을 제공하지 않도록 하는 동시에, 30초 미만의 응답으로 HTTP/1.1 캐시를 제한할 것이다.

> Note: Because the meaning of "Pragma: no-cache" in responses is not specified, it does not provide a reliable replacement for "Cache-Control: no-cache" in them.
>
> 참고: 응답에서 "Pragma: no-cache"의 의미가 명시되어 있지 않기 때문에, "Cache-Control: no-cache"의 신뢰할 수 있는 대체물을 제공하지 않는다.

5.5. Warning

The "Warning" header field is used to carry additional information about the status or transformation of a message that might not be reflected in the status code. This information is typically used to warn about possible incorrectness introduced by caching operations or transformations applied to the payload of the message.

"Warning" 헤더 필드는 상태 코드에 반영되지 않을 수 있는 메시지의 상태 또는 변환에 대한 추가 정보를 전달하기 위해 사용된다. 이 정보는 일반적으로 캐싱 작업이나 메시지의 페이로드에 적용된 변환에 의해 진행될 수 있는 부정확성에 대해 경고하기 위해 사용된다.

Warnings can be used for other purposes, both cache-related and otherwise. The use of a warning, rather than an error status code, distinguishes these responses from true failures.

경고는 캐시 관련 및 다른 목적으로 모두 사용될 수 있다. 오류 상태 코드가 아닌 경고를 사용하면 이러한 응답과 실제 실패를 구별할 수 있다.

Warning header fields can in general be applied to any message, however some warn-codes are specific to caches and can only be applied to response messages.

Warning 헤더 필드는 일반적으로 모든 메시지에 적용될 수 있지만, 일부 warn-code는 캐시에만 적용되며 응답 메시지에만 적용할 수 있다.

```
Warning       = 1#warning-value

warning-value = warn-code SP warn-agent SP warn-text
                                        [ SP warn-date ]

warn-code  = 3DIGIT
warn-agent = ( uri-host [ ":" port ] ) / pseudonym
                ; the name or pseudonym of the server adding
                ; the Warning header field, for use in debugging
                ; a single "-" is recommended when agent unknown
warn-text  = quoted-string
warn-date  = DQUOTE HTTP-date DQUOTE
```

Multiple warnings can be generated in a response (either by the origin server or by a cache), including multiple warnings with the same warn-code number that only differ in warn-text.

warn-text만 다른 동일한 warn-code를 가진 복수의 경고를 포함하여 (원서버 또는 캐시에 의해) 응답에 복수의 경고를 생성할 수 있다.

A user agent that receives one or more Warning header fields SHOULD inform the user of as many of them as possible, in the order that they appear in the response. Senders that generate multiple Warning header fields are encouraged to order them

with this user agent behavior in mind. A sender that generates new Warning header fields MUST append them after any existing Warning header fields.

하나 이상의 Warning 헤더 필드를 수신한 사용자 에이전트는 응답에 나타나는 순서대로 가능한 많은 필드를 사용자에게 알려야 한다.(SHOULD) 여러 개의 Warning 헤더 필드를 생성하는 발신자는 사용자 에이전트 동작을 염두에 두고 해당 필드를 주문할 것을 권장한다. 새 Warning 헤더 필드를 생성하는 발신자는 기존 Warning 헤더 필드 뒤에 추가해야 한다.(MUST)

Warnings are assigned three digit warn-codes. The first digit indicates whether the Warning is required to be deleted from a stored response after validation:

경고에는 세 자릿수의 warn-code가 할당된다. 첫 번째 숫자는 검토 후 저장된 응답에서 Warning을 삭제해야 하는지 여부를 나타낸다.

o 1xx warn-codes describe the freshness or validation status of the response, and so they MUST be deleted by a cache after validation. They can only be generated by a cache when validating a cached entry, and MUST NOT be generated in any other situation.

o 1xx warn-code는 응답의 신선도 또는 검토 상태를 설명하므로 검토 후 캐시에 의해 삭제되어야 한다.(MUST) 캐시된 항목을 검토 할 때만 캐시에 의해 생성될 수 있으며, 다른 상황에서는 생성되지 않아야 한다.(MUST NOT)

o 2xx warn-codes describe some aspect of the representation that is not rectified by a validation (for example, a lossy compression of the representation) and they MUST NOT be deleted by a cache after validation, unless a full response is sent, in which case they MUST be.

o 2xx warn-code는 검토에 의해 수정되지 않은 표현의 일부 측면(예, 표현의 손실 압축)을 설명하며, 전체 응답이 전송되지 않는 한 (이 경우는 삭제되어야 한다,(MUST)), 검토 후 캐시로부터 삭제되지 않아야 한다.(MUST NOT)

If a sender generates one or more 1xx warn-codes in a message to be sent to a recipient known to implement only HTTP/1.0, the sender MUST include in each corresponding warning-value a warn-date that matches the Date header field in the message. For example:

발신자가 HTTP/1.0만 구현하는 것으로 알려진 수신자에게 보낼 메시지에 하나 이상의 1xx warn-code를 생성하는 경우, 발신자는 해당 warning-value에 메시지의 Date 헤더 필드에 일치하는 warn-date를 포함해야 한다.(MUST) 예를 들어:

```
HTTP/1.1 200 OK
Date: Sat, 25 Aug 2012 23:34:45 GMT
Warning: 112 - "network down" "Sat, 25 Aug 2012 23:34:45 GMT"
```

Warnings have accompanying warn-text that describes the error, e.g., for logging. It is advisory only, and its content does not affect interpretation of the warn-code.

경고에는 오류(예: 로깅)를 설명하는 warn-text가 첨부되어 있다. 그것은 권고사항일 뿐이며, 그 내용은 warn-code 해석에 영향을 미치지 않는다.

If a recipient that uses, evaluates, or displays Warning header fields receives a warn-date that is different from the Date value in the same message, the recipient MUST exclude the warning-value containing that warn-date before storing, forwarding, or using the message. This allows recipients to exclude warning-values that were improperly retained after a cache validation. If all of the warning-values are excluded, the recipient MUST exclude the Warning header field as well.

Warning 헤더 필드를 사용, 평가 또는 표시하는 수신자는 동일한 메시지의 Date 값과 다른 warn-date를 수신하는 경우, 수신자는 메시지를 저장, 전달 또는 사용하기 전에 해당 warn-date를 포함하는 warning-value을 제외해야 한다. 따라서 수신자는 캐시 검토 후 부적절하게 보존된 warning-value을 제외할 수 있다. 모든 warning-value을 제외할 경우 수신자는 Warning 헤더 필드도 제외해야 한다.

The following warn-codes are defined by this specification, each with a recommended warn-text in English, and a description of its meaning. The procedure for defining additional warn codes is described in Section 7.2.1.

다음 warn-code는 각각 영어로 된 권장 warn-text와 그 의미에 대한 설명으로 이 명세에 의해 정의된다. 추가 경고 코드를 정의하는 절차는 Section 7.2.1에 설명되어 있다.

5.5.1. Warning: 110 - "Response is Stale"

A cache SHOULD generate this whenever the sent response is stale.

전송된 응답이 오래될 때마다 캐시가 이 값을 생성해야 한다.(SHOULD)

5.5.2. Warning: 111 - "Revalidation Failed"

A cache SHOULD generate this when sending a stale response because an attempt to validate the response failed, due to an inability to reach the server.

서버에 연결할 수 없어 응답을 검토하지 못했기 때문에 오래된 응답을 보낼 때 캐시가 이를 생성해야 한다.(SHOULD)

5.5.3. Warning: 112 - "Disconnected Operation"

A cache SHOULD generate this if it is intentionally disconnected from the rest of the network for a period of time.

캐시는 일정 기간 동안 네트워크의 나머지 부분과 의도적으로 연결이 끊어진 경우 이를 생성해야 한다.(SHOULD)

5.5.4. Warning: 113 - "Heuristic Expiration"

A cache SHOULD generate this if it heuristically chose a freshness lifetime greater than 24 hours and the response's age is greater than 24 hours.

휴리스틱적으로, 24시간보다 큰 신선도 수명을 선택하고 응답 나이가 24시간보다 큰 경우 캐시가 이를 생성해야 한다.(SHOULD)

5.5.5. Warning: 199 - "Miscellaneous Warning"

The warning text can include arbitrary information to be presented to a human user or logged. A system receiving this warning MUST NOT take any automated action, besides presenting the warning to the user.

경고 텍스트는 인간 사용자에게 표시하거나 기록할 임의의 정보를 포함할 수 있다. 이 경고를 수신하는 시스템은 사용자에게 경고를 표시하는 것 외에 자동화된 조치를 취해서는 안 된다.(MUST NOT)

5.5.6. Warning: 214 - "Transformation Applied"

This Warning code MUST be added by a proxy if it applies any transformation to the representation, such as changing the content-coding, media-type, or modifying the representation data, unless this Warning code already appears in the response.

이 Warning 코드가 응답에서 이미 나타나지 않는 한, content-coding 변경, media-type 변경 또는 표현 데이터 수정과 같은 표현에 변환을 적용하는 경우 이 Warning 코드를 프락시에 의해 추가해야 한다.(MUST)

5.5.7. Warning: 299 - "Miscellaneous Persistent Warning"

The warning text can include arbitrary information to be presented to a human user or logged. A system receiving this warning MUST NOT take any automated action.

경고 텍스트는 인간 사용자에게 표시하거나 기록할 임의의 정보를 포함할 수 있다. 이 경고를 수신하는 시스템은 자동화된 조치를 취해서는 안 된다.(MUST NOT)

6. History Lists

User agents often have history mechanisms, such as "Back" buttons and history lists, that can be used to redisplay a representation retrieved earlier in a session.

사용자 에이전트는 세션에서 이전에 검색된 표현을 다시 재생하는 데 사용할 수 있는 "뒤로" 버튼과 기록 목록과 같은 기록 메커니즘을 가지고 있는 경우가 많다.

The freshness model (Section 4.2) does not necessarily apply to history mechanisms. That is, a history mechanism can display a previous representation even if it has expired.

신선도 모델(Section 4.2)이 반드시 기록 메커니즘에 적용되는 것은 아니다. 즉, 기록 메커니즘은 만료된 경우에도 이전의 표현을 보여줄 수 있다.

This does not prohibit the history mechanism from telling the user that a view might be stale or from honoring cache directives (e.g., Cache-Control: no-store).

이는 기록 메커니즘이 사용자에게 보기가 오래되었을 수 있음을 알리거나 캐시 지시어를 준수하는 것을 금지하지 않는다(e.g., Cache-Control: no-store).

7. IANA Considerations

7.1. Cache Directive Registry

The "Hypertext Transfer Protocol (HTTP) Cache Directive Registry" defines the namespace for the cache directives. It has been created and is now maintained at <http://www.iana.org/assignments/http-cache-directives>.

"Hypertext Transfer Protocol (HTTP) Cache Directive Registry"는 캐시 지시어의 네임스페이스를 정의한다. 그것은 만들어졌고 현재 <http://www.iana.org/assignments/http-cache-directives>에서 유지되고 있다.

7.1.1. Procedure

A registration MUST include the following fields:

등록에는 다음 필드가 포함되어야 한다.

o Cache Directive Name

o Pointer to specification text

Values to be added to this namespace require IETF Review (see [RFC5226], Section 4.1).

이 네임스페이스에 추가할 값은 IETF 검토가 필요하다([RFC5226, Section 4.1 참조).

7.1.2. Considerations for New Cache Control Directives

New extension directives ought to consider defining:

새로운 확장 지시어는 다음을 정의하는 것을 고려해야 한다.

o What it means for a directive to be specified multiple times,

o 지시어가 여러 번 지정되는 것이 무엇을 의미하는지,

o When the directive does not take an argument, what it means when an argument is present,

o 지시어가 인수를 받아들이지 않을 때, 인수가 있을 때 그것이 무엇을 의미하는지,

o When the directive requires an argument, what it means when it is missing,

o 지시어가 인수를 필요할 때, 그것이 누락되었을 때 무엇을 의미하는지,

o Whether the directive is specific to requests, responses, or able to be used in either.

o 지시어가 요청, 응답 또는 어느 하나에 사용될 수 있는지 여부.

See also Section 5.2.3.
Section 5.2.3을 참조.

7.1.3. Registrations

The registry has been populated with the registrations below:

레지스트리는 아래 등록 항목으로 채워졌다.

Cache Directive	Reference
max-age	Section 5.2.1.1, Section 5.2.2.8
max-stale	Section 5.2.1.2
min-fresh	Section 5.2.1.3
must-revalidate	Section 5.2.2.1
no-cache	Section 5.2.1.4, Section 5.2.2.2
no-store	Section 5.2.1.5, Section 5.2.2.3
no-transform	Section 5.2.1.6, Section 5.2.2.4
only-if-cached	Section 5.2.1.7
private	Section 5.2.2.6
proxy-revalidate	Section 5.2.2.7
public	Section 5.2.2.5
s-maxage	Section 5.2.2.9
stale-if-error	[RFC5861], Section 4
stale-while-revalidate	[RFC5861], Section 3

7.2. Warn Code Registry

The "Hypertext Transfer Protocol (HTTP) Warn Codes" registry defines the namespace for warn codes. It has been created and is now maintained at <http://www.iana.org/assignments/http-warn-codes>.

"Hypertext Transfer Protocol (HTTP) Warn Codes" 레지스트리는 경고 코드의 네임스페이스를 정의한다. 그것은 만들어졌고 현재 <http://www.iana.org/assignments/http-warn-codes>에서 유지되고 있다.

7.2.1. Procedure

A registration MUST include the following fields:

등록에는 다음 필드가 포함되어야 한다.

- Warn Code (3 digits)
- Short Description
- Pointer to specification text

Values to be added to this namespace require IETF Review (see [RFC5226], Section 4.1).

이 네임스페이스에 추가할 값은 IETF 검토가 필요하다([RFC5226, Section 4.1 참조).

7.2.2. Registrations

The registry has been populated with the registrations below:

레지스트리는 아래 등록 항목으로 채워졌다.

Warn Code	Short Description	Reference

110	Response is Stale	Section 5.5.1
111	Revalidation Failed	Section 5.5.2
112	Disconnected Operation	Section 5.5.3
113	Heuristic Expiration	Section 5.5.4
199	Miscellaneous Warning	Section 5.5.5
214	Transformation Applied	Section 5.5.6
299	Miscellaneous Persistent Warning	Section 5.5.7

7.3. Header Field Registration

HTTP header fields are registered within the "Message Headers" registry maintained at <http://www.iana.org/assignments/message-headers/>.

HTTP 헤더 필드는 <http://www.iana.org/assignments/message-headers/>에서 유지되는 "Message Headers" 레지스트리 내에 등록된다.

This document defines the following HTTP header fields, so the "Permanent Message Header Field Names" registry has been updated accordingly (see [BCP90]).

본 문서는 다음과 같은 HTTP 헤더 필드를 정의하므로, 그에 따라 "Permanent Message Header Field Names" 레지스트리가 업데이트되었다([BCP90] 참조).

Header Field Name	Protocol	Status	Reference
Age	http	standard	Section 5.1
Cache-Control	http	standard	Section 5.2
Expires	http	standard	Section 5.3
Pragma	http	standard	Section 5.4
Warning	http	standard	Section 5.5

The change controller is: "IETF (iesg@ietf.org) - Internet Engineering Task Force".

8. Security Considerations

This section is meant to inform developers, information providers, and users of known security concerns specific to HTTP caching. More general security considerations are addressed in HTTP messaging [RFC7230] and semantics [RFC7231].

이 섹션은 HTTP 캐싱과 관련된 알려진 보안 문제를 개발자, 정보 제공자 및 사용자에게 알리기 위한 것이다. 보다 일반적인 보안 고려사항은 HTTP 메시징 [RFC7230] 및 의미론 [RFC7231]에서 다루어진다.

Caches expose additional potential vulnerabilities, since the contents of the cache represent an attractive target for malicious exploitation. Because cache contents persist after an HTTP request is complete, an attack on the cache can reveal information long after a user believes that the information has been removed from the network. Therefore, cache contents need to be protected as sensitive information.

캐시 내용은 악의적인 공격의 매력적인 대상을 나타내기 때문에 캐시는 추가적인 잠재적 취약성을 노출시킨다. 캐시 내용은 HTTP 요청이 완료된 후에도 지속되기 때문에, 사용자가 네트워크에서 정보가 제거되었다고 믿고 한참 후에 캐시에 대한 공격이 정보를 노출할 수 있다. 따라서 캐시 내용은 민감한 정보로 보호될 필요가 있다.

In particular, various attacks might be amplified by being stored in a shared cache; such "cache poisoning" attacks use the cache to distribute a malicious payload to many clients, and are especially effective when an attacker can use implementation flaws, elevated privileges, or other techniques to insert such a response into a cache. One common attack vector for cache poisoning is to exploit differences in message parsing on proxies and in user agents; see Section 3.3.3 of [RFC7230] for the relevant requirements.

특히, 다양한 공격은 공유 캐시에 저장되어 증폭될 수 있다. 그러한 "캐시 중독" 공격은 많은 클라이언트에 악의적인 페이로드 배포를 위해 캐시를 사용하며, 특히 공격자가 구현 결함, 높은 권한 또는 기타 기술을 사용하여 그러한 응답을 캐시에 삽입할 수 있을 때 효과적이다. 캐시 중독에 대한 하나의 일반적인 공격 벡터는 프락시 및 사용자 에이전트의 메시지 구문 분석의 차이를 이용하는 것이다. 관련 요건은 [RFC7230]의 Section 3.3.3을 참조한다.

Likewise, implementation flaws (as well as misunderstanding of cache operation) might lead to caching of sensitive information (e.g., authentication credentials) that is thought to be private, exposing it to unauthorized parties.

마찬가지로, 구현 결함(캐시 작동에 대한 오해뿐만 아니라)은 비공개적이라고 생각되는 민감한 정보(e.g., 인증 자격 증명)를 캐싱하여 이를 허가받지 않은 당사자에게 노출시킬 수 있다.

Furthermore, the very use of a cache can bring about privacy concerns. For example, if two users share a cache, and the first one browses to a site, the second may be able to detect that the other has been to that site, because the resources from it load more quickly, thanks to the cache.

게다가, 캐시를 사용하는 것은 사생활에 대한 우려를 불러 일으킬 수 있다. 예를 들어, 두 명의 사용자가 캐시를 공유하고 첫 번째 사용자가 사이트를 탐색할 경우, 두 번째 사용자가 캐시 덕분에 다른 사용자가 해당 사이트에 갔다는 것을 탐지할 수 있을 것이다.

Note that the Set-Cookie response header field [RFC6265] does not inhibit caching; a cacheable response with a Set-Cookie header field can be (and often is) used to satisfy subsequent requests to caches. Servers who wish to control caching of these responses are encouraged to emit appropriate Cache-Control response header fields.

Set-Cookie 응답 헤더 필드 [RFC6265]는 캐싱을 금지하지 않으며, Set-Cookie 헤더 필드가 있는 캐시 가능한 응답은 후속 캐시 요청을 충족하는 데 사용될 수 있으며 자주 사용된다는 점에 유의하십시오. 이러한 응답의 캐시를 제어하려는 서버는 적절한 Cache-Control 응답 헤더 필드를 내보내는 것이 좋다.

9. Acknowledgments

See Section 10 of [RFC7230].

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, June 2014.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, June 2014.

[RFC7232] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests", RFC 7232, June 2014.

[RFC7233] Fielding, R., Ed., Lafon, Y., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Range Requests", RFC 7233, June 2014.

[RFC7235] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Authentication", RFC 7235, June 2014.

10.2. Informative References

[BCP90] Klyne, G., Nottingham, M., and J. Mogul, "Registration Procedures for Message Header Fields", BCP 90, RFC 3864, September 2004.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5861] Nottingham, M., "HTTP Cache-Control Extensions for Stale Content", RFC 5861, April 2010.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

[RFC6265] Barth, A., "HTTP State Management Mechanism", RFC 6265, April 2011.

Appendix A. Changes from RFC 2616

The specification has been substantially rewritten for clarity.

명세서는 명확성을 위해 실질적으로 다시 작성되었다.

The conditions under which an authenticated response can be cached have been clarified. (Section 3.2)

인증된 응답을 캐시할 수 있는 조건이 명확해졌다. (Section 3.2)

New status codes can now define that caches are allowed to use heuristic freshness with them. Caches are now allowed to calculate heuristic freshness for URIs with query components. (Section 4.2.2)

새로운 상태 코드는 이제 캐시가 휴리스틱한 신선도를 함께 사용할 수 있도록 허용된다는 것을 정의할 수 있다. 캐시는 이제 쿼리 구성요소가 있는 URI의 휴리스틱한 신선도를 계산할 수 있다. (Section 4.2.2)

The algorithm for calculating age is now less conservative. Caches are now required to handle dates with time zones as if they’re invalid, because it's not possible to accurately guess. (Section 4.2.3)

나이를 계산하는 알고리즘은 이제 덜 보수적이다. 캐시는 이제 정확한 추측이 불가능하기 때문에 시간대가 있는 날짜를 무효인 것처럼 다루어야 한다. (Section 4.2.3)

The Content-Location response header field is no longer used to determine the appropriate response to use when validating. (Section 4.3)

Content-Locaion 응답 헤더 필드는 더 이상 유효성 검사 시 사용할 적절한 응답을 결정하는 데 사용되지 않는다. (Section 4.3)

The algorithm for selecting a cached negotiated response to use has been clarified in several ways. In particular, it now explicitly allows header-specific canonicalization when processing selecting header fields. (Section 4.1)

사용할 캐시된 협상 응답 선택 알고리즘은 여러 가지 방법으로 명확히 하였다. 특히 이제는 헤더 필드 선택 처리 시 헤더별 표준화를 명시적으로 허용하고 있다. (Section 4.1)

Requirements regarding denial-of-service attack avoidance when performing invalidation have been clarified. (Section 4.4)

무효화 수행 시 denial-of-service 공격 회피에 관한 요구사항이 명확해졌다. (Section 4.4)

Cache invalidation only occurs when a successful response is received. (Section 4.4)

캐시 무효화는 성공적인 응답이 수신된 경우에만 발생한다. (Section 4.4)

Cache directives are explicitly defined to be case-insensitive. Handling of multiple instances of cache directives when only one is expected is now defined. (Section 5.2)

캐시 지시어는 대소문자를 구분하지 않도록 명시적으로 정의되어 있다. 한 개만 예상될 때 캐시 지시어의 여러 인스턴스를 처리하는 것이 이제는 정의된다. (Section 5.2)

The "no-store" request directive doesn't apply to responses; i.e., a cache can satisfy a request with no-store on it and does not invalidate it. (Section 5.2.1.5)

"no-store" 요청 지시어는 응답에 적용되지 않는다. 즉, 캐시는 no-store 요청을 충족할 수 있고 무효화하지 않는다. (Section 5.2.1.5)

The qualified forms of the private and no-cache cache directives are noted to not be widely implemented; for example, "private=foo" is interpreted by many caches as

simply "private". Additionally, the meaning of the qualified form of no-cache has been clarified. (Section 5.2.2)

private 캐시 및 no-cache 캐시 지시어 적격 형태는 널리 구현되지 않는 것으로 알려져 있다. 예를 들어, "private=foo"는 단순한 "private"로 해석되는 많은 캐시가 있다. 또한, no-cache의 자격 있는 형태의 의미가 명확해졌다. (Section 5.2.2)

The "no-cache" response directive's meaning has been clarified. (Section 5.2.2.2)

"no-cache" 응답 지시어의 의미가 명확해졌다. (Section 5.2.2)

The one-year limit on Expires header field values has been removed; instead, the reasoning for using a sensible value is given. (Section 5.3)

Expires 헤더 필드 값에 대한 1년 제한은 제거되었다. 대신, 합리적인 값을 사용하기 위한 추론이 제시된다. (Section 5.3)

The Pragma header field is now only defined for backwards compatibility; future pragmas are deprecated. (Section 5.4)

Pragma 헤더 필드는 이제 역호환성만을 위해 정의되며, 미래에 pragma 더 이상 사용되지 않는다. (Section 5.4)

Some requirements regarding production and processing of the Warning header fields have been relaxed, as it is not widely implemented. Furthermore, the Warning header field no longer uses RFC 2047 encoding, nor does it allow multiple languages, as these aspects were not implemented. (Section 5.5)

Warning 헤더 필드의 생산 및 처리와 관련된 일부 요구사항은 광범위하게 구현되지 않았기 때문에 완화되었다. 또한 Warning 헤더 필드는 더 이상 RFC 2047 인코딩을 사용하지 않으며, 이러한 측면이 구현되지 않았기 때문에 여러 언어를 허용하지 않는다. (Section 5.5)

This specification introduces the Cache Directive and Warn Code Registries, and defines considerations for new cache directives. (Section 7.1 and Section 7.2)

이 명세는 Cache 지시어 및 Warn Code Registries를 도입하고, 새로운 Cache 지시어에 대한 고려사항을 정의한다. (Section 7.1 및 Section 7.2)

Appendix B. Imported ABNF

The following core rules are included by reference, as defined in Appendix B.1 of [RFC5234]: ALPHA (letters), CR (carriage return), CRLF (CR LF), CTL (controls), DIGIT (decimal 0-9), DQUOTE (double quote), HEXDIG (hexadecimal 0-9/A-F/a-f), LF (line feed), OCTET (any 8-bit sequence of data), SP (space), and VCHAR (any visible US-ASCII character).

The rules below are defined in [RFC7230]:

```
OWS            = <OWS, see [RFC7230], Section 3.2.3>
field-name     = <field-name, see [RFC7230], Section 3.2>
quoted-string  = <quoted-string, see [RFC7230], Section 3.2.6>
token          = <token, see [RFC7230], Section 3.2.6>

port           = <port, see [RFC7230], Section 2.7>
pseudonym      = <pseudonym, see [RFC7230], Section 5.7.1>
uri-host       = <uri-host, see [RFC7230], Section 2.7>
```

The rules below are defined in other parts:

```
HTTP-date      = <HTTP-date, see [RFC7231], Section 7.1.1.1>
```

Appendix C. Collected ABNF

In the collected ABNF below, list rules are expanded as per Section 1.2 of [RFC7230].

```
Age = delta-seconds

Cache-Control = *( "," OWS ) cache-directive *( OWS "," [ OWS cache-directive ] )

Expires = HTTP-date

HTTP-date = <HTTP-date, see [RFC7231], Section 7.1.1.1>
```

OWS = <OWS, see [RFC7230], Section 3.2.3>

Pragma = *("," OWS) pragma-directive *(OWS "," [OWS pragma-directive])

Warning = *("," OWS) warning-value *(OWS "," [OWS warning-value])

cache-directive = token ["=" (token / quoted-string)]

delta-seconds = 1*DIGIT

extension-pragma = token ["=" (token / quoted-string)]

field-name = <field-name, see [RFC7230], Section 3.2>

port = <port, see [RFC7230], Section 2.7>
pragma-directive = "no-cache" / extension-pragma
pseudonym = <pseudonym, see [RFC7230], Section 5.7.1>

quoted-string = <quoted-string, see [RFC7230], Section 3.2.6>

token = <token, see [RFC7230], Section 3.2.6>

uri-host = <uri-host, see [RFC7230], Section 2.7>

warn-agent = (uri-host [":" port]) / pseudonym
warn-code = 3DIGIT
warn-date = DQUOTE HTTP-date DQUOTE
warn-text = quoted-string
warning-value = warn-code SP warn-agent SP warn-text [SP warn-date]