Transfer-Encoding header

- Syntax
- Directives
- Examples
  - Response with chunked encoding
- Specifications
- Browser compatibility
- See also

The HTTP **`Transfer-Encoding`** request and response header specifies the form of encoding used to transfer messages between nodes on the network.

`Transfer-Encoding` is a hop-by-hop header, that is applied to a message between two nodes, not to a resource itself. Each segment of a multi-node connection can use different `Transfer-Encoding` values. If you want to compress data over the whole connection, use the end-to-end Content-Encoding header instead.

In practice this header is rarely used, and in those cases it is almost always used with `chunked`.

That said, the specification indicates that when present in a message it indicates the compression used on the message in that hop, and/or whether the message has been chunked. For example, `Transfer-Encoding: gzip, chunked` indicates that the content has been compressed using the gzip coding and then chunked using the chunked coding while forming the message body.

The header is optional in responses to a HEAD request as these messages have no body and, therefore, no transfer encoding. When present it indicates the value that would have applied to the corresponding response to a GET message, if that `GET` request did not include a preferred `Transfer-Encoding`.

> **WARNING:** HTTP/2 disallows all uses of the `Transfer-Encoding` header. HTTP/2 and later provide more efficient mechanisms for data streaming than chunked transfer. Usage of the header in HTTP/2 may likely result in a specific `protocol`

> `error`.

Header type	[Request header](https://developer.mozilla.org/en-US/docs/Glossary/Request%20header), [Response header](https://developer.mozilla.org/en-US/docs/Glossary/Response%20header), [Content header](https://developer.mozilla.org/en-US/docs/Glossary/Content%20header)
[Forbidden request header](https://developer.mozilla.org/en-US/docs/Glossary/Forbidden%20request%20header)	Yes

Syntax

```
Transfer-Encoding: chunked
Transfer-Encoding: compress
Transfer-Encoding: deflate
Transfer-Encoding: gzip

// Several values can be listed, separated by a comma
Transfer-Encoding: gzip, chunked
```

Directives

- `chunked`
  - : Data is sent in a series of chunks. Content can be sent in streams of unknown size to be transferred as a sequence of length-delimited buffers, so the sender can keep a connection open, and let the recipient know when it has received the entire message. The Content-Length header must be omitted, and at the beginning of each chunk, a string of hex digits indicate the size of the chunk-data in octets, followed by `\r\n` and then the chunk itself, followed by another `\r\n`. The terminating chunk is a zero-length chunk.
- `compress`
  - : A format using the Lempel-Ziv-Welch (LZW) algorithm. The value name was taken from the UNIX *compress* program, which implemented this algorithm. Like the compress program, which has disappeared from most UNIX distributions, this content-encoding is used by almost no browsers today, partly because of a patent issue (which expired in 2003).

- `deflate`
  - : Using the zlib structure (defined in RFC 1950), with the *deflate* compression algorithm (defined in RFC 1951).
- `gzip`
  - : A format using the Lempel-Ziv coding (LZ77), with a 32-bit CRC. This is originally the format of the UNIX *gzip* program. The HTTP/1.1 standard also recommends that the servers supporting this content-encoding should recognize `x-gzip` as an alias, for compatibility purposes.

Examples

Response with chunked encoding

Chunked encoding is useful when larger amounts of data are sent to the client and the total size of the response may not be known until the request has been fully processed. For example, when generating a large HTML table resulting from a database query or when transmitting large images. A chunked response looks like this:

```
HTTP/1.1 200 OK
Content-Type: text/plain
Transfer-Encoding: chunked

7\r\n
Welcome\r\n
1c\r\n
to Mozilla Developer Network\r\n
0\r\n
\r\n
```

Specifications

Browser compatibility

See also

- Accept-Encoding
- Content-Encoding
- Content-Length
- Chunked transfer encoding

□□□ #3
□ □□□ □□□ 2026-10-31 03:27:53 UTC
□ cocobook-sync □□□ 2026-11-04 17:36:08 UTC