1.3 HTTP

Introduction to HTTP

The ***HyperText Transfer Protocol (HTTP)*** is a networking protocol that runs over TCP/IP and governs communication between web browsers and web servers. ***Transmission Control Protocol/Internet Protocol (TCP/IP)*** is a protocol suite that governs how data packets are transferred over the internet from one machine to another. Understanding the details of TCP/IP is not usually required of web developers, but a thorough understanding of HTTP is necessary to create effective web applications.

HTTP versions

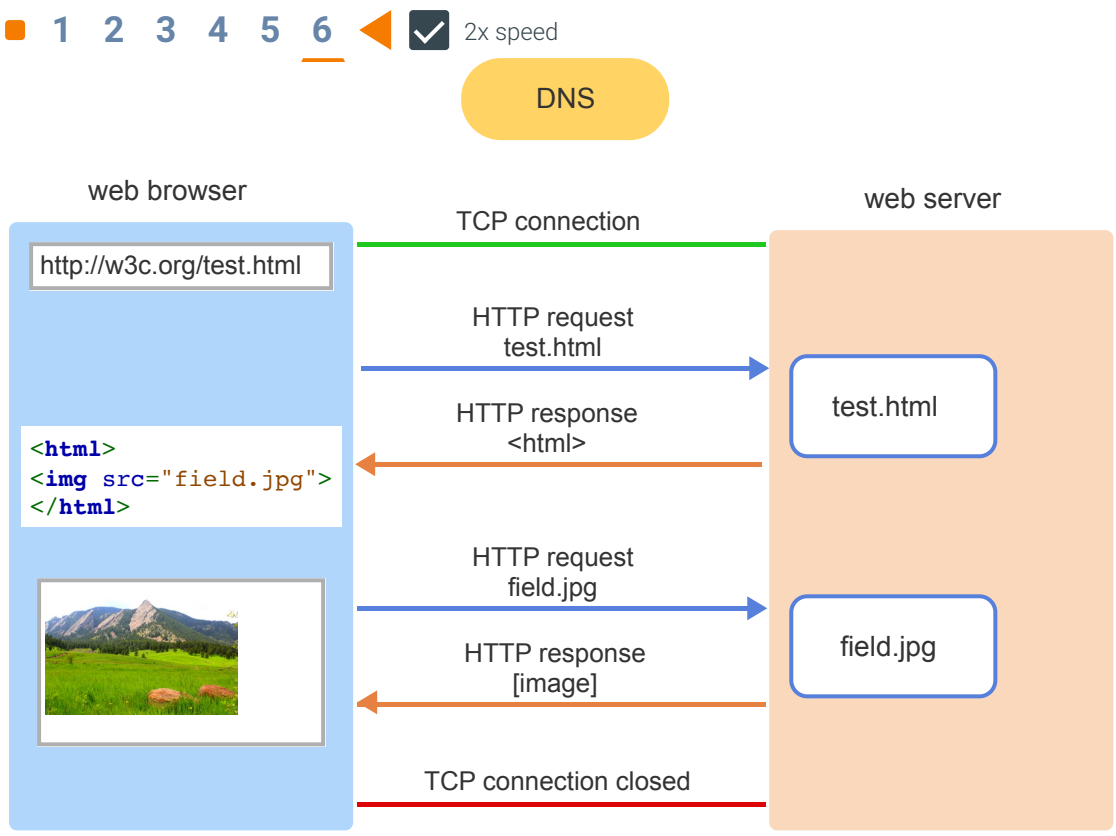
HTTP/1.1 *is the HTTP standard used for most of the web's lifetime, but many websites are adopting* ***HTTP/2****, a relatively new HTTP standard that speeds-up the transfer of information between web browsers and web servers. HTTP/2 maintains most of HTTP/1.1's semantics.* ***HTTP/3****, currently in development, improves the speed of HTTP/2 by using UDP to transport data packets instead of TCP. This material focuses on the basic HTTP workings that all standards share.*

Before HTTP communication begins, the web browser extracts the domain name from the URL being accessed and performs a DNS lookup. The web browser performs a ***DNS lookup*** by sending the domain name to the local DNS and getting back the IP address of the web server hosting the domain name. Ex: https://www.w3c.org/ has a domain name of w3c.org, and DNS translates w3c.org to the IP address 193.51.208.66. The web browser uses the IP address to establish a TCP connection with the web server and begins communicating with HTTP.

HTTP functions as a request-response protocol between web browsers and web servers:

- An ***HTTP request*** is a message sent from the web browser to the web server. Often the request asks the web server to send back a web resource like an HTML file, image, CSS stylesheet, JavaScript file, or video.
- An ***HTTP response*** is a message sent from the web server back to the web browser in response to an HTTP request. Often the response contains the requested web resource.

PARTICIPATION ACTIVITY 1.3.1: HTTP requests and responses.



When no more resources are needed, the browser closes the TCP connection.

Captions

1. The web browser does a DNS lookup of the domain name w3c.org and uses the IP address to create a TCP connection to the web server.
2. The web browser sends an HTTP request asking for the resource test.html.
3. The web server returns the contents of test.html to the browser in an HTTP response.
4. The browser parses the HTML and sends a second HTTP request for field.jpg.
5. The web server returns the image to the browser in an HTTP response. The browser displays the image in the webpage.
6. When no more resources are needed, the browser closes the TCP connection.

Feedback?

PARTICIPATION ACTIVITY 1.3.2: HTTP requests and responses.

1) Before a TCP connection is created, the web browser usually performs a/an _____.

- HTTP request
- HTTP response
- DNS lookup

Correct

The browser performs a DNS lookup to obtain the web server's IP address. The browser uses the IP address to establish a TCP connection with the web server.

2) How many total HTTP requests does a browser send for a webpage that does not use any other web resources?

- 1
- 2
- 3

Correct

Only one HTTP request is made for the HTML file.

3) How many total HTTP requests does a browser send for a webpage that contains four web resources: an image, a video, a CSS stylesheet, and a JavaScript file?

- 1
- 4
- 5

Correct

One HTTP request is made for the HTML file, and one request is made for each resource. The browser may send multiple requests in parallel to speed-up the transfer of resources.

4) If a web browser sends 20 HTTP requests to a web server, how many HTTP responses will the web server likely send to the web browser?

- 0
- 10
- 20

Correct

Every HTTP request usually results in a single HTTP response. However, a web server can ignore a request and fail to send a response.

Feedback?

Request and response headers

An HTTP request and an HTTP response are both composed of four parts:

1. Start line - The ***start line*** specifies the HTTP version being used. A request's start line includes a request type and path; a response's start line includes a status code and phrase.
2. Zero or more header fields - A ***header field*** is a keyword followed by a colon and a value. Header fields supply additional information about the request or response.
3. A blank line
4. Optional message body - A ***message body*** contains data being transferred between a web browser and web server. In a request, the message body may be empty or contain submitted form data. In a response, the message body may contain the requested resource.

The figures below show an example HTTP request for a Wikipedia URL and the corresponding HTTP response containing HTML.

Figure 1.3.1: HTTP request with no message body.

Request for URL: https://en.wikipedia.org/wiki/World_Wide_Web

```
GET /wiki/World_Wide_Web HTTP/1.1
Host: en.wikipedia.org
User-Agent: Mozilla/5.0 Chrome/48.0.2564
```

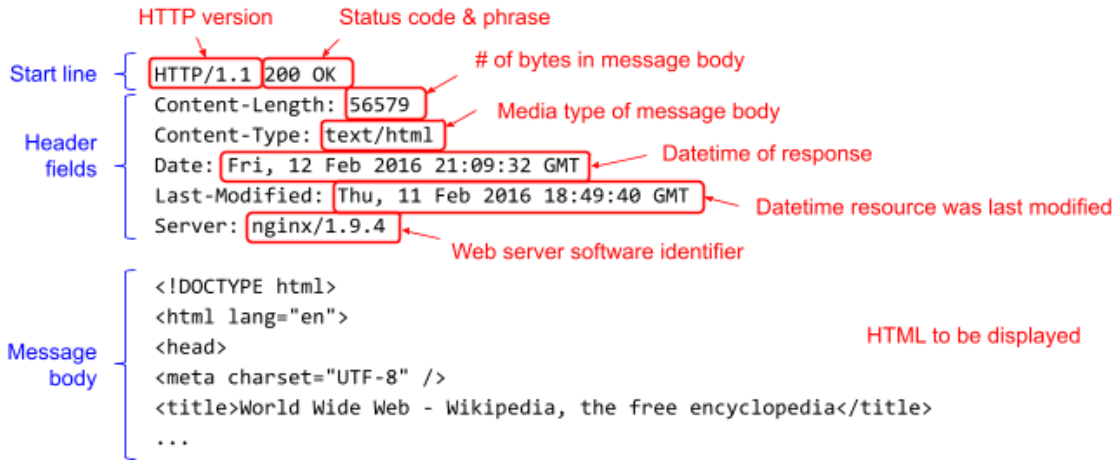
Feedback?

Figure 1.3.2: HTTP response with HTML in message body.

Response for URL: https://en.wikipedia.org/wiki/World_Wide_Web

```
HTTP/1.1 200 OK
Content-Length: 56579
Content-Type: text/html
Date: Fri, 12 Feb 2016 21:09:32 GMT
Last-Modified: Thu, 11 Feb 2016 18:49:40 GMT
Server: nginx/1.9.4

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<title>World Wide Web - Wikipedia, the free encyclopedia</title>
...
```



Feedback?

PARTICIPATION ACTIVITY 1.3.3: HTTP request and response headers.

If unable to drag and drop, refresh the page.

Content-Length	Number of bytes in the response's message body. The web browser uses the content length to determine how much data to expect from the web server.	Correct
	Media type of the response's message body. The media type tells the browser what type of resource to expect in the body. Media types are known	Correct

Content-Type	as *MIME types*. A MIME type has a type (category) and a subtype (specific type), separated by a slash. Common MIME types include: text/html, text/css, image/jpeg, application/pdf, and video/quicktime.	
Date	Datetime the response was generated by the web server. Datetime values in HTTP headers are represented in Greenwich Mean Time (GMT), which is equal to Coordinated Universal Time (UTC).	Correct
Last-Modified	Datetime the requested resource was last modified on the web server. The last modified datetime is usually available for static files on the web server. For resources that are dynamically created by the web server, the last modified datetime is often not provided.	Correct
Server	Identifies the web server software that generated the response. Popular values for Server include Apache, Microsoft-IIS, and nginx.	Correct
Host	The domain name for the requested path. Some websites are hosted at the same IP address, so the Host field helps the web server determine which website's resource is being requested.	Correct
User-Agent	Identifies the browser making the request. Browsers often identify themselves using a unique string of characters. The User-Agent field may be used by a web server to respond back with different resources depending on which browser made the request. Ex: A	Correct

desktop page may be sent back to a desktop Chrome browser, but a mobile webpage may be sent back to an iPhone browser.

Reset

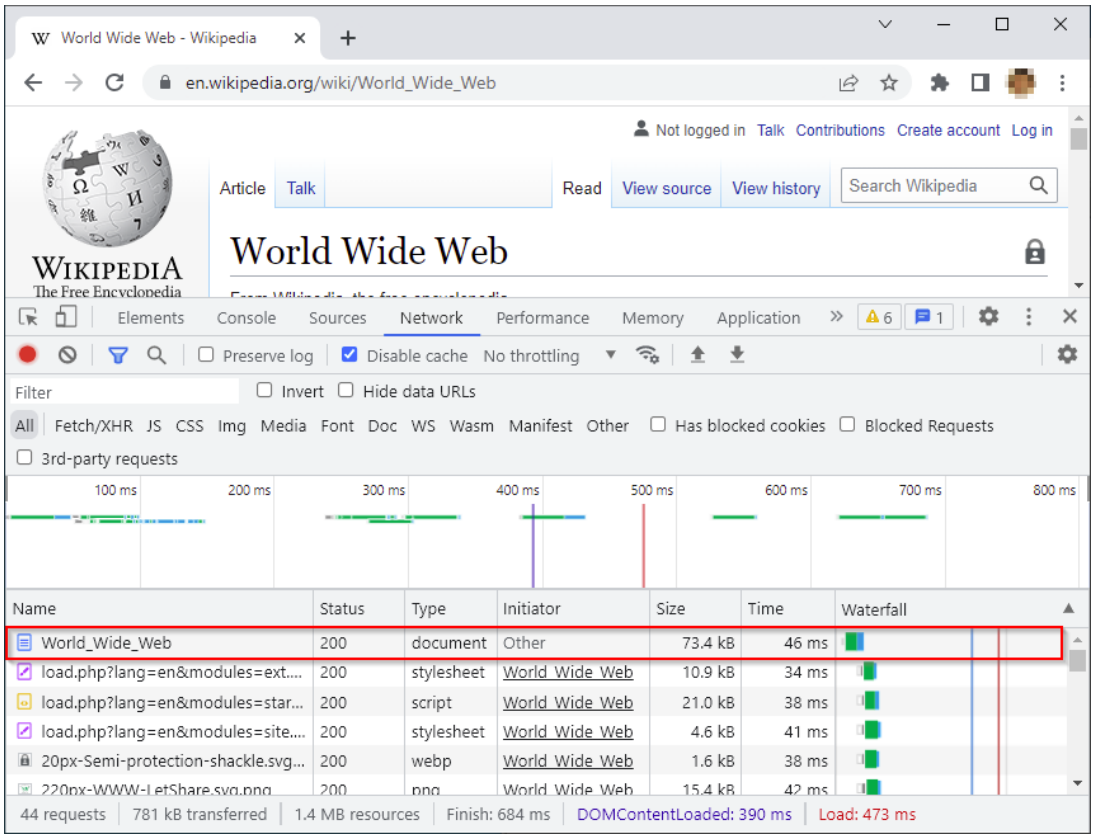
Feedback?

List of HTTP headers

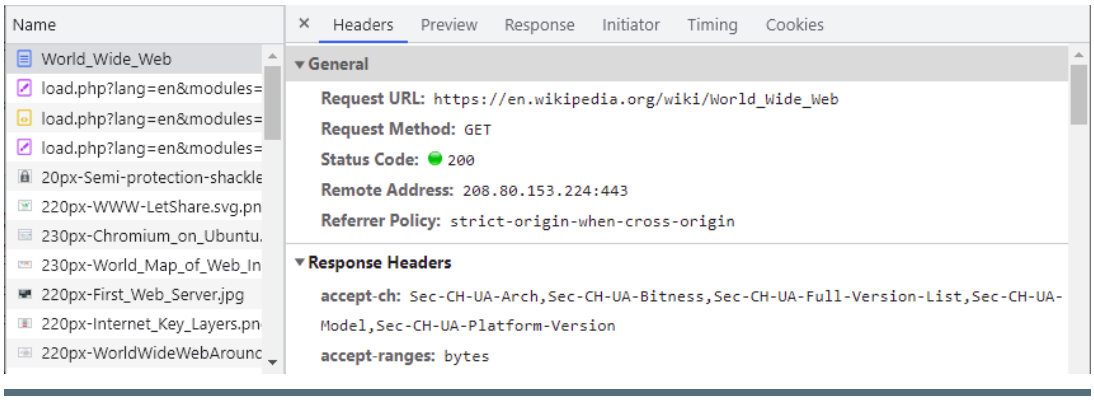
*The **Internet Assigned Numbers Authority** (**IANA**) is a standards organization that manages various internet numbers and symbols, like global IP address allocation, root zone management in DNS, and media types. IANA maintains a list of HTTP headers that are currently active, obsolete, or experimental.*

Chrome DevTools for watching HTTP traffic.

All popular web browsers contain built-in developer tools. Developers can access ***Chrome's developer tools*** (***DevTools***) by pressing Ctrl+Shift+I (Windows) or Command-Option-I (Mac). In the figure below, the Network tab shows the HTTP network traffic when accessing Wikipedia's article on World Wide Web. The first HTTP request is highlighted. All subsequent requests are for other resources used in the webpage.



Clicking on a specific request shows the request and response details.



Request methods and response status codes

The most common HTTP request-response scenario is when the web browser issues a request with the GET request method, and the web server returns the requested resource with a 200 status code. An HTTP ***request method*** indicates the desired action to perform on a resource. Other request methods besides GET may also be sent in an HTTP request. Ex: POST is often used when the web browser is sending information from a web form to the web server. The POST, PATCH, PUT, and DELETE request methods are used by web services that allow resources to be created, modified, and deleted on the web server.

Table 1.3.1: Common HTTP request methods.

Request method	Meaning
GET	Request a representation of the specified resource.
HEAD	Request a response identical to GET but without the response body.
POST	Create a new resource with the contents of the message body.
PATCH	Modify an existing resource with the contents of the message body.
PUT	Replace an existing resource with the contents of the message body.
DELETE	Delete an existing resource.

Feedback?

An HTTP response ***status code*** is a three digit number that indicates the status of the requested resource. A successfully requested resource results in a 200 status code, and other status codes are returned for various reasons. Ex: A 301 or 302 status code redirects the browser to a different URL. A ***browser redirect*** is when the web server returns a 301 or 302 status code with a `Location` header indicating the URL the browser should load next.

Table 1.3.2: Common HTTP response status codes.

Status code	Status phrase	Meaning
200	OK	Standard response for a successful request.
301	Moved Permanently	The resource should always be requested at a different URL.
302	Found	The resource should temporarily be requested at a different URL.
304	Not Modified	The resource has not been modified since the last time the resource was requested.
403	Forbidden	The web browser does not have permission to access the resource.

Status code	Status phrase	Meaning
404	Not Found	The resource could not be located.
500	Internal Server Error	Something unexpected happened on the web server.

Feedback?

URL shortening

URL shortening *is a technique to create shorter URLs that redirect to longer URLs. Ex: http://en.wikipedia.org/wiki/URL_shortening has a short URL of http://tinyurl.com/urlwiki. Short URLs are convenient for sharing on social media, especially on Twitter where the number of characters in a post is limited. Common URL shortening services include bit.ly, rebrandly.com, and tinyurl.com.*

When a user types or clicks on a short URL, the URL shortening service responds with a 301 status code and a Location header with the webpage's full URL. The example HTTP request and response below shows http://tinyurl.com/urlwiki redirects to http://en.wikipedia.org/wiki/URL_shortening.

HTTP request	*HTTP response*

`GET /urlwiki HTTP/1.1`
`Host: tinyurl.com`
`User-Agent: Mozilla/5.0 Chrome/48.0.2564`
	`HTTP/1.1 301 Moved Permanently`
	`Date: Tue, 16 Feb 2016 16:38:59 GMT`
	`Location: http://en.wikipedia.org/wiki/URL_shortening`

PARTICIPATION ACTIVITY 1.3.4: HTTP request methods and response status codes.

1) Which request method is used by the browser to get the same response headers that a GET would generate but without the message body?

- GET
- HEAD
- PUT

Correct

HEAD only requests the response headers.

2) Which request method is used by the browser to submit web form data to the web server?

- GET
- PUT
- POST

Correct

POST is the request method most often used to submit web form data to the web server.

3) Which status code is sent when the web server wants to redirect the browser to a different URL?

- 200
- 301 or 302
- 403

Correct

301 and 302 status codes are always accompanied by a Location header that indicates the new URL that the browser should access. Ex: "Location: http://w3c.org/new-location.html". 302 status codes are for URLs that may not redirect in the future.

4) Which status code is sent when the requested URL does not point to an existing resource on the web server?

- 200
- 301
- 404

Correct

404 can occur when the user mistypes a URL, a URL for a resource has changed, or a resource has been removed.

5) Which status code is sent when the web browser is denied permission to the requested URL?

- 301
- 304
- 403 (selected)

Correct

403 frequently occurs when the URL corresponds to a directory that does not have directory listing permissions or a file that doesn't have read permissions.

Feedback?

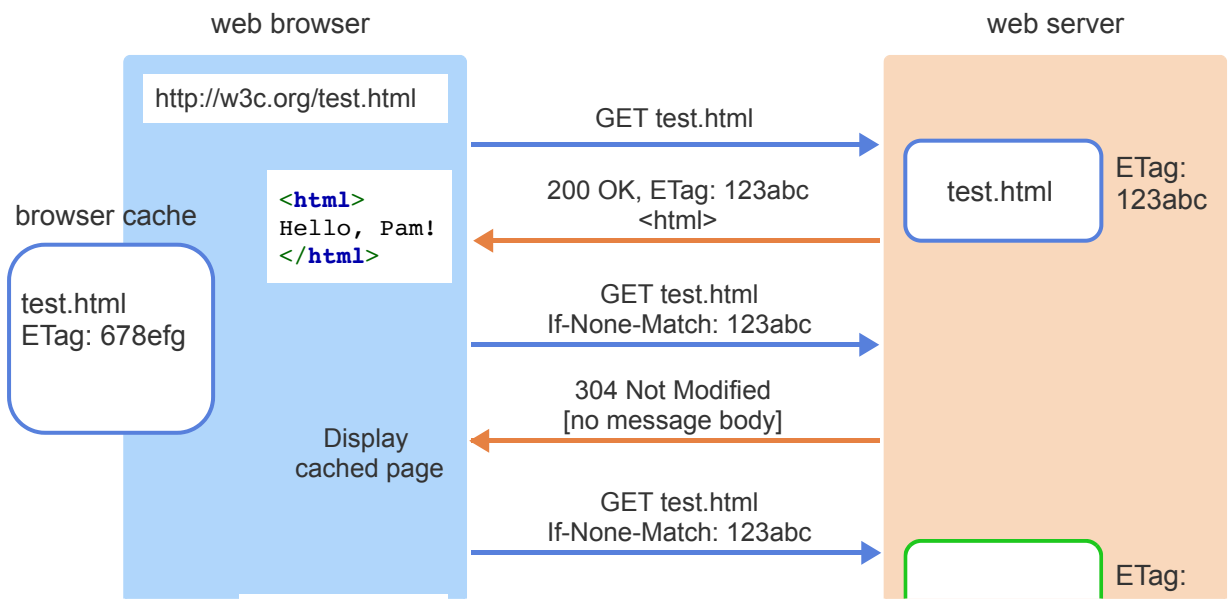
Browser caching

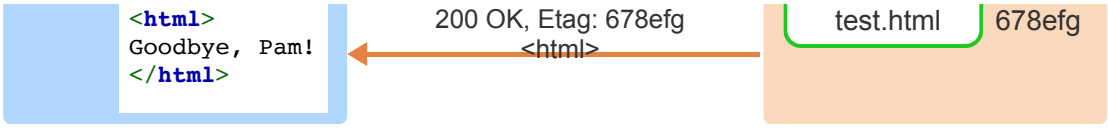
Most web browsers use a browser cache to store requested content. A ***browser cache*** is an area on the computer's file system where web content can be stored by the web browser for quick retrieval later. By caching web content, browsers can reduce the amount of network traffic required to display previously visited webpages. Ex: If a webpage is accessed now and again ten minutes from now, the browser can display the cached webpage instead of re-downloading the webpage. If the webpage has changed in the 10 minute span, the browser should download the updated page.

Web browsers often use ETags to aid in caching web resources. An ***entity tag (ETag)*** is an identifier for a specific version of a web resource. Ex: 34905a3e285dd11. When the resource changes, so should the ETag associated with the resource. When a web browser requests a cached web resource, the browser sends the ETag in the request with an `If-None-Match` header. The web server will reply with a 304 Not Modified response status if the resource has not changed or a 200 OK with the changed resource and a new ETag.

PARTICIPATION ACTIVITY 1.3.5: Requesting cached resources with ETags.

1 2 3 4 5 6 7 2x speed





The browser displays the new content and updates the browser cache.

Captions

1. The browser requests uncached resource test.html. The web server returns the contents of test.html back to the browser where test.html is cached.
2. The second request for cached test.html includes the ETag 123abc.
3. The web server compares ETags and notes the file has not changed, so 304 is returned with no HTML. The browser displays the cached content.
4. test.html is modified on the web server and assigned a new ETag.
5. The browser sends a third request for test.html with ETag 123abc.
6. The web server compares ETags and notes the file has changed, so 200 is returned with new HTML and ETag.
7. The browser displays the new content and updates the browser cache.

Feedback?

PARTICIPATION ACTIVITY 1.3.6: Requesting cached content.

1) Which status code is sent when the web server compares the cached ETag with the server's ETag for a requested web resource, and the two ETags are identical?

- 200
- 304
- 403

Correct

When the ETags are equal, the web browser has an up-to-date cached copy of the web resource.

2) Does a 304 response generally contain a message body?

- Yes
- No

Correct

304 indicates the cached content is up-to-date, so re-sending the same content to the browser is unnecessary.

3) Do web servers always generate ETags for all web resources?

- ○ Yes
- ◉ No

Correct

ETags are not often produced for dynamic web content.

4) Where can the HTTP header field `If-None-Match` be found?

- ◉ HTTP request
- ○ HTTP response

Correct

The browser sends the If-None-Match header field with the ETag to the web server in the HTTP request.

Feedback?

Other caching mechanisms

HTTP defines additional HTTP headers to aid in caching:

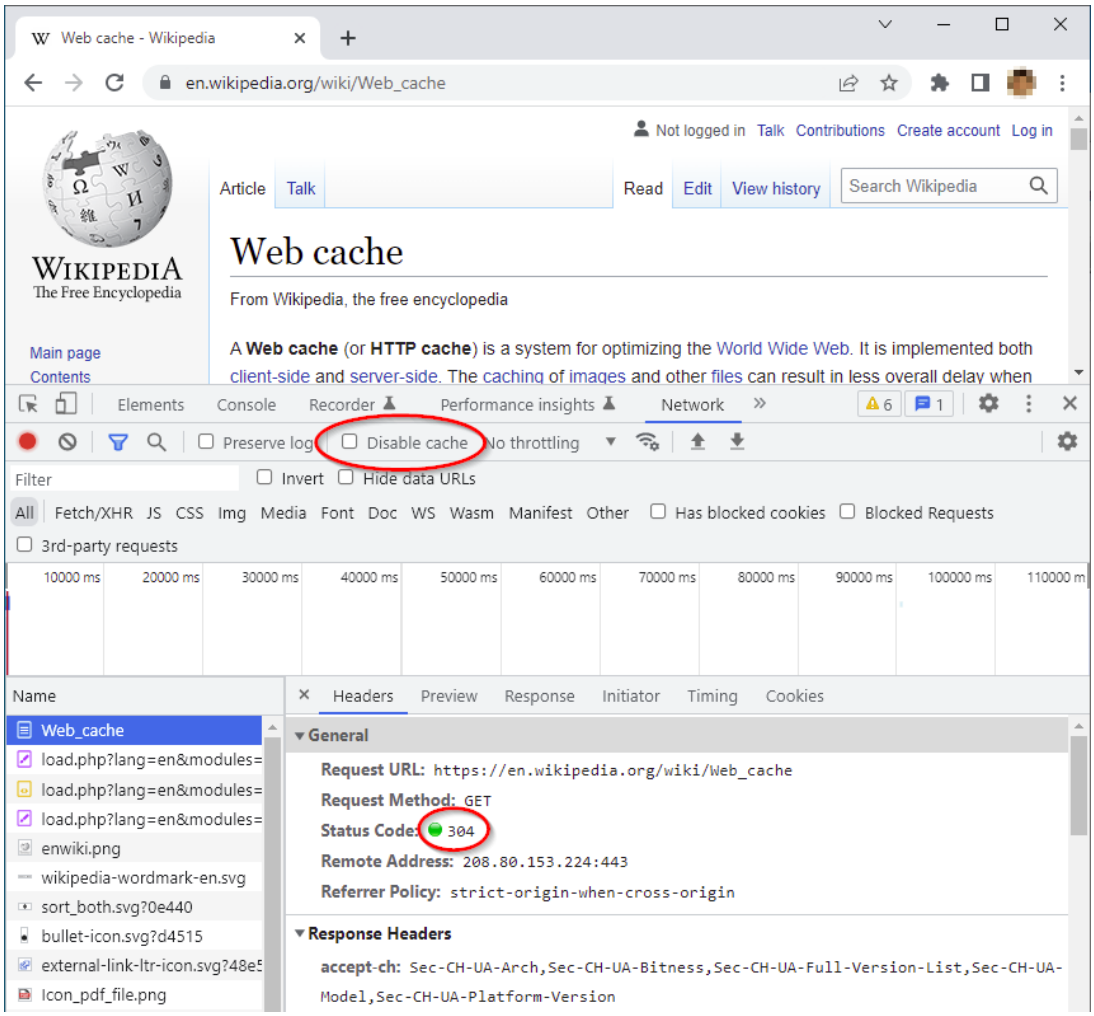
- ***If-Modified-Since*** *is used with the* `Last-Modified` *date/time to request the web server only send the requested resource if the resource has changed since the specified date/time. Ex: "If-Modified-Since: Wed, 01 Sep 2019 13:24:52 GMT" asks the web server to send the resource if the resource was modified after Sep 1, 2019 at 13:24:52 GMT.*
- ***Expires*** *contains a date/time indicating when the requested resource is considered "stale". Ex: "Expires: Wed, 01 Sep 2019 13:24:52 GMT" tells the web browser to show the cached resource until Sep 1, 2019 at 13:24:52 GMT.*
- ***Cache-Control*** *is used to specify a number of caching directives. Ex: "Cache-Control: no-store" tells the web browser to never cache the requested resource, and "Cache-Control: max-age=180" tells the browser to cache the resource for 180 seconds.*

Viewing 304 responses in Chrome DevTools

Web developers can view 304 Not Modified responses in Chrome's DevTools by opening the Network tab and ensuring the checkbox labeled "Disable cache" is

not checked. If "Disable cache" is checked then nothing will be cached, and `If-None-Match` *and* `If-Modified-Since` *headers will not be sent.*

The Wikipedia page about web cache is a good page to test in DevTools to see 304 responses. When the developer loads the page once, the web server returns a 200 status code for the Web_cache request. When the developer presses the browser's Reload button, the server returns a 304 status code.

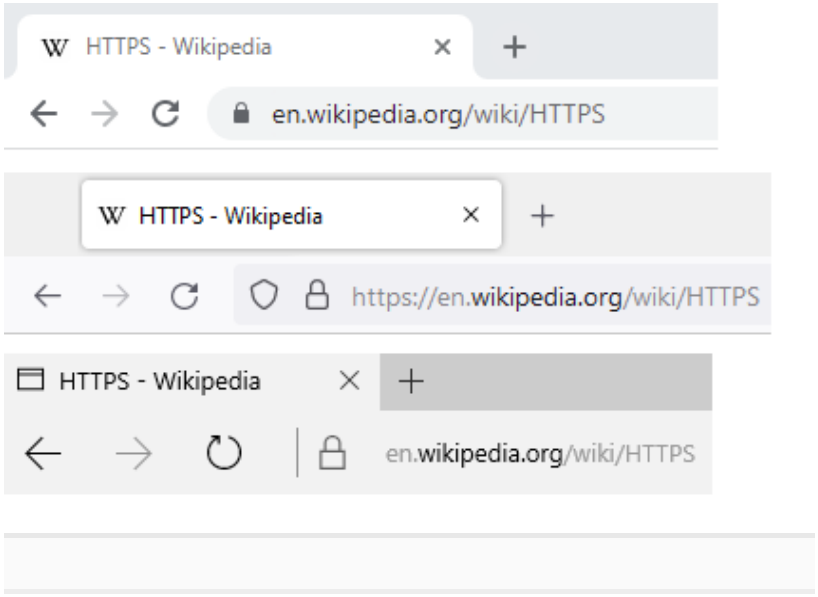


HTTPS

All HTTP traffic can be viewed by third parties using a network sniffer. A ***network sniffer*** is software that monitors network traffic and allows users to inspect HTTP requests and responses. ***HTTPS*** encrypts HTTP traffic between a browser and web server so a network sniffer cannot intercept sensitive information in the HTTP traffic like passwords, credit card numbers, financial transactions, etc.

HTTPS uses a protocol called ***Transport Layer Security*** (***TLS***), which uses asymmetric public keys to encrypt data between the browser and web server. A website wanting to use HTTPS must acquire a ***digital certificate***, issued by a trusted ***certificate authority***, that contains a public key used by TLS to encrypt data.

Figure 1.3.3: Chrome, Firefox, and Edge browsers showing a padlock symbol when HTTPS is used.



Feedback?

PARTICIPATION ACTIVITY 1.3.7: Steps in an HTTPS transaction.

Put the steps used in an HTTPS transaction in order.

If unable to drag and drop, refresh the page.

Browser requests an HTTPS connection to a webpage.	Step 1 The URL must begin with "https://".	Correct
Web server sends digital certificate to the browser.	Step 2 The browser warns the user if the digital certificate is not from a trusted certificate authority.	Correct
Browser and web server generate session keys.	Step 3 The session keys are used to encrypt and decrypt data.	Correct
Browser and web server transmit encrypted data.	Step 4 The encrypted data can only be decrypted by the browser and web	Correct

server.

Reset

Feedback?

Exploring further:

- HTTP response status codes (MDN)
- ETag - HTTP (MDN)
- Moving to HTTPS Guide
- Inspect Network Activity with Chrome DevTools
- HTTP/2
- HTTP/3

CHALLENGE ACTIVITY 1.3.1: HTTP.

530096.4000608.qx3zqy7

Jump to level 1

1 2 3 4

A web server contains resource cities.html with an ETag of 678pqr.

Indicate the status code returned and the ETag stored in the cache if:

the browser requests cities.html with the header If-None-Match: 678pqr.

status code: 304 ETag: 678pqr

the browser requests cities.html with the header If-None-Match: 234ijk.

status code: 200 ETag: 678pqr

1 2 3 4

Check Next

Done. Click any level to practice more. Completion is preserved.

✔ Expected:
304, 678pqr
200, 678pqr

If the request for cached cities.html includes the ETag 678pqr, then the web server compares ETags and notes that the file has not changed. Therefore, 304 is returned HTML. The ETag in the cache remains 678pqr.

If the request for cached cities.html includes the ETag 234ijk, then the web server cc ETags and notes that the file has changed. Therefore, 200 is returned with new HTM ETag. The ETag in the cache is updated to 678pqr.

Feedback?

How was this section? **Provide section feedback**