Hypertext Transfer Protocol (HTTP) response status codes

Here is a list of HTTP response status codes.

1xx Informational - Request received, continuing process. This class of status code indicates a provisional response, consisting only of the Status-Line and optional headers, and is terminated by an empty line. Since HTTP/1.0 did not define any 1xx status codes, servers must not send a 1xx response to an HTTP/1.0 client except under experimental conditions.

- 100 Continue - This means that the server has received the request headers, and that the client should proceed to send the request body (in the case of a request for which a body needs to be sent; for example, a POST request). If the request body is large, sending it to a server when a request has already been rejected based upon inappropriate headers is inefficient. To have a server check if the request could be accepted based on the request's headers alone, a client must send Expect: 100-continue as a header in its initial request and check if a 100 Continue status code is received in response before continuing (or receive 417 Expectation Failed and not continue).
- 101 Switching Protocols - This means the requester has asked the server to switch protocols and the server is acknowledging that it will do so.
- 102 Processing (WebDAV; RFC 2518) - As a WebDAV request may contain many sub-requests involving file operations, it may take a long time to complete the request. This code indicates that the server has received and is processing the request, but no response is available yet. This prevents the client from timing out and assuming the request was lost.

2xx Success - This class of status codes indicates the action requested by the client was received, understood, accepted and processed successfully.

- 200 OK - Standard response for successful HTTP requests. The actual response will depend on the request method used. In a GET request, the response will contain an entity corresponding to the requested resource. In a POST request the response will contain an entity describing or containing the result of the action.
- 201 Created - The request has been fulfilled and resulted in a new resource being created.
- 202 Accepted - The request has been accepted for processing, but the processing has not been completed. The request might or might not eventually be acted upon, as it might be disallowed when processing actually takes place.
- 203 Non-Authoritative Information (since HTTP/1.1) - The server successfully processed the request, but is returning information that may be from another source.
- 204 No Content - The server successfully processed the request, but is not returning any content. Usually used as a response to a successful delete request.
- 205 Reset Content - The server successfully processed the request, but is not returning any content. Unlike a 204 response, this response requires that the requester reset the document view.
- 206 Partial Content - The server is delivering only part of the resource (byte serving) due to a range header sent by the client. The range header is used by tools like wget to enable resuming of interrupted downloads, or split a download into multiple simultaneous streams.
- 207 Multi-Status (WebDAV; RFC 4918) - The message body that follows is an XML message and can contain a number of separate response codes, depending on how many sub-requests were

made.
- 208 Already Reported (WebDAV; RFC 5842) - The members of a DAV binding have already been enumerated in a previous reply to this request, and are not being included again.
- 226 IM Used (RFC 3229) - The server has fulfilled a request for the resource, and the response is a representation of the result of one or more instance-manipulations applied to the current instance.

3xx Redirection - This class of status code indicates the client must take additional action to complete the request. Many of these status codes are used in URL redirection. A user agent may carry out the additional action with no user interaction only if the method used in the second request is GET or HEAD. A user agent should not automatically redirect a request more than five times, since such redirections usually indicate an infinite loop.

- 300 Multiple Choices - Indicates multiple options for the resource that the client may follow. It, for instance, could be used to present different format options for video, list files with different extensions, or word sense disambiguation.
- 301 Moved Permanently - This and all future requests should be directed to the given URI.
- 302 Found - This is an example of industry practice contradicting the standard. The HTTP/1.0 specification (RFC 1945) required the client to perform a temporary redirect (the original describing phrase was “Moved Temporarily”), but popular browsers implemented 302 with the functionality of a 303 See Other. Therefore, HTTP/1.1 added status codes 303 and 307 to distinguish between the two behaviours. However, some Web applications and frameworks use the 302 status code as if it were the 303.
- 303 See Other (since HTTP/1.1) - The response to the request can be found under another URI using a GET method. When received in response to a POST (or PUT/DELETE), it should be assumed that the server has received the data and the redirect should be issued with a separate GET message.
- 304 Not Modified - Indicates that the resource has not been modified since the version specified by the request headers If-Modified-Since or If-None-Match. This means that there is no need to retransmit the resource, since the client still has a previously-downloaded copy.
- 305 Use Proxy (since HTTP/1.1) - The requested resource is only available through a proxy, whose address is provided in the response. Many HTTP clients (such as Mozilla and Internet Explorer) do not correctly handle responses with this status code, primarily for security reasons.
- 306 Switch Proxy - No longer used. Originally meant “Subsequent requests should use the specified proxy.”
- 307 Temporary Redirect (since HTTP/1.1) - In this case, the request should be repeated with another URI; however, future requests should still use the original URI. In contrast to how 302 was historically implemented, the request method is not allowed to be changed when reissuing the original request. For instance, a POST request should be repeated using another POST request.
- 308 Permanent Redirect (Experimental RFC; RFC 7238) - The request, and all future requests should be repeated using another URI. 307 and 308 (as proposed) parallel the behaviours of 302 and 301, but do not allow the HTTP method to change. So, for example, submitting a form to a permanently redirected resource may continue smoothly.

4xx Client Error - The 4xx class of status code is intended for cases in which the client seems to have errored. Except when responding to a HEAD request, the server should include an entity containing an explanation of the error situation, and whether it is a temporary or permanent condition. These status codes are applicable to any request method. User agents should display any included entity to the user.

- 400 Bad Request - The server cannot or will not process the request due to something that is perceived to be a client error.
- 401 Unauthorized - Similar to 403 Forbidden, but specifically for use when authentication is required and has failed or has not yet been provided. The response must include a WWW-Authenticate header field containing a challenge applicable to the requested resource. See Basic access authentication and Digest access authentication.
- 402 Payment Required - Reserved for future use. The original intention was that this code might be used as part of some form of digital cash or micropayment scheme, but that has not happened, and this code is not usually used. YouTube uses this status if a particular IP address has made excessive requests, and requires the person to enter a CAPTCHA.
- 403 Forbidden - The request was a valid request, but the server is refusing to respond to it. Unlike a 401 Unauthorized response, authenticating will make no difference.
- 404 Not Found - The requested resource could not be found but may be available again in the future. Subsequent requests by the client are permissible.
- 405 Method Not Allowed - A request was made of a resource using a request method not supported by that resource; for example, using GET on a form which requires data to be presented via POST, or using PUT on a read-only resource.
- 406 Not Acceptable - The requested resource is only capable of generating content not acceptable according to the Accept headers sent in the request.
- 407 Proxy Authentication Required - The client must first authenticate itself with the proxy.
- 408 Request Timeout - The server timed out waiting for the request. According to HTTP specifications: “The client did not produce a request within the time that the server was prepared to wait. The client MAY repeat the request without modifications at any later time.”
- 409 Conflict - Indicates that the request could not be processed because of conflict in the request, such as an edit conflict in the case of multiple updates.
- 410 Gone - Indicates that the resource requested is no longer available and will not be available again. This should be used when a resource has been intentionally removed and the resource should be purged. Upon receiving a 410 status code, the client should not request the resource again in the future. Clients such as search engines should remove the resource from their indices. Most use cases do not require clients and search engines to purge the resource, and a “404 Not Found” may be used instead.
- 411 Length Required - The request did not specify the length of its content, which is required by the requested resource.
- 412 Precondition Failed - The server does not meet one of the preconditions that the requester put on the request.
- 413 Request Entity Too Large - The request is larger than the server is willing or able to process.
- 414 Request-URI Too Long - The URI provided was too long for the server to process. Often the result of too much data being encoded as a query-string of a GET request, in which case it should be converted to a POST request.
- 415 Unsupported Media Type - The request entity has a media type which the server or resource does not support. For example, the client uploads an image as image/svg+xml, but the server requires that images use a different format.
- 416 Requested Range Not Satisfiable - The client has asked for a portion of the file (byte serving), but the server cannot supply that portion. For example, if the client asked for a part of the file that lies beyond the end of the file.
- 417 Expectation Failed - The server cannot meet the requirements of the Expect request-header field.
- 418 I'm a teapot (RFC 2324) - This code was defined in 1998 as one of the traditional IETF April Fools' jokes, in RFC 2324, Hyper Text Coffee Pot Control Protocol, and is not expected to be implemented by actual HTTP servers.

- 419 Authentication Timeout (not in RFC 2616) - Not a part of the HTTP standard, 419 Authentication Timeout denotes that previously valid authentication has expired. It is used as an alternative to 401 Unauthorized in order to differentiate from otherwise authenticated clients being denied access to specific server resources.
- 420 Method Failure (Spring Framework) - Not part of the HTTP standard, but defined by Spring in the HttpStatus class to be used when a method failed. This status code is deprecated by Spring.
- 420 Enhance Your Calm (Twitter) - Not part of the HTTP standard, but returned by version 1 of the Twitter Search and Trends API when the client is being rate limited. Other services may wish to implement the 429 Too Many Requests response code instead.
- 422 Unprocessable Entity (WebDAV; RFC 4918) - The request was well-formed but was unable to be followed due to semantic errors.
- 423 Locked (WebDAV; RFC 4918) - The resource that is being accessed is locked.
- 424 Failed Dependency (WebDAV; RFC 4918) - The request failed due to failure of a previous request (e.g., a PROPPATCH).
- 426 Upgrade Required - The client should switch to a different protocol such as TLS/1.0.
- 428 Precondition Required (RFC 6585) - The origin server requires the request to be conditional. Intended to prevent “the 'lost update' problem, where a client GETs a resource's state, modifies it, and PUTs it back to the server, when meanwhile a third party has modified the state on the server, leading to a conflict.”
- 429 Too Many Requests (RFC 6585) - The user has sent too many requests in a given amount of time. Intended for use with rate limiting schemes.
- 431 Request Header Fields Too Large (RFC 6585) - The server is unwilling to process the request because either an individual header field, or all the header fields collectively, are too large.
- 440 Login Timeout (Microsoft) - A Microsoft extension. Indicates that your session has expired.
- 444 No Response (Nginx) - Used in Nginx logs to indicate that the server has returned no information to the client and closed the connection (useful as a deterrent for malware).
- 449 Retry With (Microsoft) - A Microsoft extension. The request should be retried after performing the appropriate action. Often search-engines or custom applications will ignore required parameters. Where no default action is appropriate, the Aviongoo website sends a “HTTP/1.1 449 Retry with valid parameters: param1, param2, . . .” response. The applications may choose to learn, or not.
- 450 Blocked by Windows Parental Controls (Microsoft) - A Microsoft extension. This error is given when Windows Parental Controls are turned on and are blocking access to the given webpage.
- 451 Unavailable For Legal Reasons (Internet draft) - Defined in the internet draft “A New HTTP Status Code for Legally-restricted Resources”. Intended to be used when resource access is denied for legal reasons, e.g. censorship or government-mandated blocked access. A reference to the 1953 dystopian novel Fahrenheit 451, where books are outlawed.
- 451 Redirect (Microsoft) - Used in Exchange ActiveSync if there either is a more efficient server to use or the server cannot access the users' mailbox. The client is supposed to re-run the HTTP Autodiscovery protocol to find a better suited server.
- 494 Request Header Too Large (Nginx) - Nginx internal code similar to 431 but it was introduced earlier in version 0.9.4 (on January 21, 2011).
- 495 Cert Error (Nginx) - Nginx internal code used when SSL client certificate error occurred to distinguish it from 4XX in a log and an error page redirection.
- 496 No Cert (Nginx) - Nginx internal code used when client didn't provide certificate to distinguish it from 4XX in a log and an error page redirection.
- 497 HTTP to HTTPS (Nginx) - Nginx internal code used for the plain HTTP requests that are sent

to HTTPS port to distinguish it from 4XX in a log and an error page redirection.
- 498 Token expired/invalid (Esri) - Returned by ArcGIS for Server. A code of 498 indicates an expired or otherwise invalid token.
- 499 Client Closed Request (Nginx) - Used in Nginx logs to indicate when the connection has been closed by client while the server is still processing its request, making server unable to send a status code back.
- 499 Token required (Esri) - Returned by ArcGIS for Server. A code of 499 indicates that a token is required (if no token was submitted).

5xx Server Error - The server failed to fulfil an apparently valid request. Response status codes beginning with the digit “5” indicate cases in which the server is aware that it has encountered an error or is otherwise incapable of performing the request. Except when responding to a HEAD request, the server should include an entity containing an explanation of the error situation, and indicate whether it is a temporary or permanent condition. Likewise, user agents should display any included entity to the user. These response codes are applicable to any request method.

- 500 Internal Server Error - A generic error message, given when an unexpected condition was encountered and no more specific message is suitable.
- 501 Not Implemented - The server either does not recognize the request method, or it lacks the ability to fulfil the request. Usually this implies future availability (e.g., a new feature of a web-service API).
- 502 Bad Gateway - The server was acting as a gateway or proxy and received an invalid response from the upstream server.
- 503 Service Unavailable - The server is currently unavailable (because it is overloaded or down for maintenance). Generally, this is a temporary state.
- 504 Gateway Timeout - The server was acting as a gateway or proxy and did not receive a timely response from the upstream server.
- 505 HTTP Version Not Supported - The server does not support the HTTP protocol version used in the request.
- 506 Variant Also Negotiates (RFC 2295) - Transparent content negotiation for the request results in a circular reference.
- 507 Insufficient Storage (WebDAV; RFC 4918) - The server is unable to store the representation needed to complete the request.
- 508 Loop Detected (WebDAV; RFC 5842) - The server detected an infinite loop while processing the request (sent in lieu of 208 Already Reported).
- 509 Bandwidth Limit Exceeded (Apache bw/limited extension) - This status code is not specified in any RFCs. Its use is unknown.
- 510 Not Extended (RFC 2774) - Further extensions to the request are required for the server to fulfil it.
- 511 Network Authentication Required (RFC 6585) - The client needs to authenticate to gain network access. Intended for use by intercepting proxies used to control access to the network (e.g., “captive portals” used to require agreement to Terms of Service before granting full Internet access via a Wi-Fi hotspot).
- 520 Origin Error (CloudFlare) - This status code is not specified in any RFCs, but is used by CloudFlare's reverse proxies to signal an “unknown connection issue between CloudFlare and the origin web server” to a client in front of the proxy.
- 521 Web server is down (CloudFlare) - This status code is not specified in any RFCs, but is used by CloudFlare's reverse proxies to indicate that the origin webserver refused the connection.
- 522 Connection timed out (CloudFlare) - This status code is not specified in any RFCs, but is used by CloudFlare's reverse proxies to signal that a server connection timed out.

- 523 Proxy Declined Request (CloudFlare) - This status code is not specified in any RFCs, but is used by CloudFlare's reverse proxies to signal a resource that has been blocked by the administrator of the website or proxy itself.
- 524 A timeout occurred (CloudFlare) - This status code is not specified in any RFCs, but is used by CloudFlare's reverse proxies to signal a network read timeout behind the proxy to a client in front of the proxy.
- 598 Network read timeout error (Unknown) - This status code is not specified in any RFCs, but is used by Microsoft HTTP proxies to signal a network read timeout behind the proxy to a client in front of the proxy.
- 599 Network connect timeout error (Unknown) - This status code is not specified in any RFCs, but is used by Microsoft HTTP proxies to signal a network connect timeout behind the proxy to a client in front of the proxy.