HTTP Status Codes

A comprehensive cheat sheet covering HTTP status codes, providing a quick reference for developers and network administrators. Includes common codes, their meanings, and practical examples.

Informational & Success Codes

1xx Informational

Informational status codes indicate that the request was received and understood. It is issued on a provisional basis while request processing continues. Alerts the client to wait for a final response.

100 Continue: The server has received the request headers, and the client should proceed to send the request body.

101 Switching Protocols: The server is switching protocols as requested by the client.

102 Processing: The server is processing the request, but has not yet completed. The server MUST send an interim response to prevent the client from timing out.

2xx Success

These status codes indicate that the client's request was successfully received, understood, and accepted.

200 OK: Standard response for successful HTTP requests. The actual response will depend on the request method used.

Example: `GET` request successful.

201 Created: The request has been fulfilled, resulting in a new resource being created.

Example: `POST` request creating a new user.

202 Accepted: The request has been accepted for processing, but the processing has not been completed.

Example: Request to process a large image.

204 No Content: The server successfully processed the request, but is not returning any content.

Example: `DELETE` request successful, no content returned.

206 Partial Content: The server is delivering only part of the resource due to a range header sent by the client.

Example: Streaming video content.

Redirection Codes

3xx Redirection

Redirection messages indicate that the client must take additional action to complete the request. Many of these status codes are used in URL redirection.

300 Multiple Choices: The server has multiple options for the resource that the client could follow. It could be a list of different file types or languages.

Example: Offering different language versions of a webpage.

301 Moved Permanently: This and all future requests should be directed to the given URI.

Example: Website moved to a new domain.

302 Found: The resource resides temporarily under a different URI. The client should continue to use the original URI for future requests.

Note: Often misused as a 'temporary' redirect, should typically use 307 or 303.

303 See Other: The response to the request can be found under another URI using a GET method.

Example: After a `POST` request, redirecting to a confirmation page.

304 Not Modified: Indicates that the resource has not been modified since the last version specified by the client. The client can continue to use the cached version.

Example: Browser cache validation.

307 Temporary Redirect: The target resource resides temporarily under a different URI and the client MUST NOT change the request method when performing the redirect.

Example: Redirecting to a maintenance page.

308 Permanent Redirect: The target resource has been assigned a new permanent URI and any future references to this resource should use one of the returned URIs.

Example: API endpoint moved permanently.

Client Error Codes

4xx Client Errors

Client error responses indicate that the request contains bad syntax or cannot be fulfilled. These are errors caused by the client, not the server.

400 Bad Request: The server cannot or will not process the request due to something that is perceived to be a client error.

Example: Invalid JSON format in request body.

401 Unauthorized: Authentication is required and has failed or has not yet been provided. The client must authenticate itself to get the requested response.

Example: Accessing a protected API endpoint without a valid token.

403 Forbidden: The client does not have access rights to the content; that is, it is unauthorized, so the server is refusing to give the requested resource. Unlike 401, the client's identity is known to the server.

Example: Accessing a resource that requires specific permissions.

404 Not Found: The server cannot find the requested resource. This status code is often used to disguise that the server refuses to fulfill the request or does not want to reveal the existence of the resource.

Example: Requesting a non-existent page.

405 Method Not Allowed: The method specified in the request is not allowed for the resource identified by the request URI.

Example: Using `POST` on a read-only endpoint.

408 Request Timeout: The server timed out waiting for the request.

Example: Slow client connection, server closes connection.

413 Payload Too Large: Request entity is larger than limits defined by server; the server might close the connection or return a Retry-After header field.

Example: Uploading a file exceeding the maximum allowed size.

429 Too Many Requests: The user has sent too many requests in a given amount of time ("rate limiting").

Example: API rate limit exceeded.

Server Error Codes

5xx Server Errors

The server has failed to fulfill a request. These responses are often due to issues on the server side.

500 Internal Server Error: A generic error message, given when an unexpected condition was encountered and no more specific message is suitable.

Example: Unhandled exception in server-side code.

501 Not Implemented: The server either does not recognize the request method, or it lacks the ability to fulfill the request. Usually this implies future availability (e.g., a new feature of a server).

Example: Requesting an unsupported feature.

502 Bad Gateway: The server was acting as a gateway or proxy and received an invalid response from the upstream server.

Example: Upstream server down or returning errors.

503 Service Unavailable: The server is not ready to handle the request. Common causes are a server that is down for maintenance or that is overloaded. Note that together with this response, a user-friendly page explaining the problem should be sent.

Example: Server undergoing maintenance.

504 Gateway Timeout: The server was acting as a gateway or proxy and did not receive a timely response from the upstream server.

Example: Upstream server taking too long to respond.

505 HTTP Version Not Supported: The server does not support the HTTP protocol version that was used in the request.

Example: Client using an outdated HTTP version.

509 Bandwidth Limit Exceeded: The server has exceeded the bandwidth allocated.

Example: Hosting plan limits reached.