

CommonHTTPStatusCodes

Here is a list of most of the used http status codes with an explanation of what they mean.

Code	Message	Description	See also
	1xx - Informational		
100	Continue	Allows the client to determine if the origin server is willing to accept the request, for example based on the request headers, before the client sends the request body. If the request is not accepted, the status code 406 is returned.	

Example usage: http client checks if chunked transfer encoding is allowed before sending the full request body. | |

101	Switching Protocols	Response from the httpd when an upgrade header is received and the requested protocol is supported by the httpd. Will be followed by a Upgrade header specifying the tokens of the protocol stack it is switching to.	
-----	---------------------	---	--

Example usage: http client wants to switch from HTTP/1.1 to TLS/1.0 in order to set up a secure communication channel. | |

202	Processing	Part of the http status code extensions for Distributed Authoring (WebDAV). Should be sent to the http client when a request takes a long period to complete. Will always be followed by the final response indicating either success or failure of the request. This status code can be used to avoid connection timeouts preventing the response from being successfully sent to the http client.	RFC 2518 Section 10
	2xx - Successful		
200	OK	Request has succeeded, content delivered normally.	
201	Created	Request has succeeded and has resulted in a new resource being created. Usually details on the resource such as its new URI and ETag information are included in the httpd response.	

Example usage: a POST request to a remote script that generates a file on the server returns the status code 201 once the file has been successfully created. | |

202	Accepted	The request has been accepted for processing, but the processing has not been completed. This is used when a client sends a request to the httpd, but can or does not maintain a persistent connection long enough to await the final response. The response containing the 202 status code should also contain an entity describing a location where the status of the request can be monitored or some estimate of when the the request can be expected to be fulfilled.	
203	Non-Authortative Information	The returned metainformation in the entity-header is not the definitive set as available from the origin server, but is gathered from a local or a third-party copy. The set presented MAY be a subset or superset of the original version. For example, including local annotation information about the resource might result in a superset of the metainformation known by the origin server. Use of this response code is not required and is only appropriate when the response would otherwise be 200.	
204	No Content	The server has fulfilled the request but does not need to return an entity-body, and might want to return updated metainformation. This response is primarily intended to allow input for actions to take place without causing a change to the user agent's active document view.	
205	Reset Content	The server has fulfilled the request and the user agent should reset the document view which caused the request to be sent. This response is primarily intended to allow input for actions to take place via user input, followed by a clearing of the form in which the input is given so that the user can easily initiate another input action.	
206	Partial Content	The server has fulfilled the partial get request for the resource. The request must have included a Range header field indicating the desired range, and may have included an If-Range header field to make the request conditional.	
207	Multi-Status	Part of the http status code extensions for Distributed Authoring (WebDAV). Used when a request to the httpd resulted in multiple non-1xx status codes being generated during the method invocation. The response body consists of a XML element 'multistatus' which contains multiple 'response' elements, each specifying a status code.	RFC 2518 Section 10
	3xx - Redirection		
300	Multiple Choices	The requested resource corresponds to any one of a set of representations, meaning Content Negotiation is in effect, on Apache HTTPD Server enabled by the Multiviews option or a type-map file.	Content Negotiation
301	Moved Permanently	The requested resource has been assigned a new permanent URI.	

Example usage: Redirect permanent /foo <http://www.example.com/bar> or RewriteRule /foo <http://www.example.com/bar> [R=301] | |

<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="0d0db3fb-3ab1-4d9c-9d80-b03a6d2ef753"><ac:plain-text-body><![CDATA[	302	Found	The requested resource resides temporarily under a different URI. This is the default status code returned if the request matches a Redirect or [RedirectMatch] directive, or a rewriterule flagged with [R.]></ac:plain-text-body></ac:structured-macro>
--	-----	-------	---

Example usage: Redirect /foo <http://www.example.com/bar> or RewriteRule /foo <http://www.example.com/bar> [R] | |

303	See Other	The response to the request can be found under a different URI and should be retrieved using a GET method on that resource. This method exists primarily to allow the output of a POST-activated script to redirect the user agent to a selected resource.	
-----	-----------	--	--

Example usage: Redirect seeother /foo <http://www.example.com/bar> or RewriteRule /foo <http://www.example.com/bar> [R=303] |

304	Not Modified	If the client has performed a conditional GET request and access is allowed, but the document has not been modified, the server should respond with this status code. The httpd response will include header fields informing the http client about the requested resource. If this information is identical to the cached version of the resource, this cached version will be used by the http client. Header fields that include such information are: Date, ETag, Content-Location, Cache-Control, Expires, and/or Vary.	Cacheing
305	Use Proxy	The requested resource must be accessed through the proxy, which is given by the Location field in the httpd response. Usually the http client will immediately request the resource described in this Location field.	mod_proxy
306		Was used in earlier versions of the RFC, currently unused.	
307	Temporary Redirect	Defined in the HTTP/1.1 specification as a more strict version of 302. See the RFC for details.	

Example usage: Redirect 307 /foo <http://www.example.com/bar> or RewriteRule /foo <http://www.example.com/bar> [R=307] | [RFC2616 Section 10](#) |

	4xx - Client Error		
400	Bad Request	The request could not be understood by the server due to malformed syntax. Usually this is because of a error in the http client itself or a plugin/addon/malware loaded in the http client. You can inspect what request was being sent exactly by looking in the access logs. When using mod_security, check if the request is matching a rule and being blocked. Very rarely caused by problems on the network layer, either client- or serverside, resulting in corrupted http packets. Note that there is a bug in IE5 and IE6 causing Bad Request errors to occur when using digest authentication. Apache HTTPD Server provides a workaround for this.	
401	Unauthorized	When the http client requests a resource that requires authentication, http status code 401 is returned. The http client will need to provide valid authentication details in the request in order to access the resource. If the http client already sent the authentication details and the status code 401 is still returned, this means the authentication details were invalid. Possibly also caused by misconfigured authentication directives or invalid or corrupted password files.	Authentication
402	Payment Required	This status code is reserved for future use, currently not returned by Apache HTTPD Server.	
403	Forbidden	The server understood the request, but is refusing to fulfill it. Common causes are:	

- * No index file and indexing is turned off
- * Deny rule matching the request
- * Resource cannot be modified (the default response to a PUT request)
- * Filesystem permission problems ([\[Logs\]](#))

Always check the errorlog for what exactly went wrong.] |

404	Not Found	The server has not found anything matching the request URI. This is commonly caused by a erroneous hyperlink or when resources are moved without modifying references to them. Could also be caused by an invalid internal redirect or rewriterule, check the accesslog and/or rewritelog to see what resource was requested exactly. Note that *nix operating systems are case-sensitive by default, so a request for /Foo while /foo was intended, will result in a 404 status code being returned.	
405	Method Not Allowed		
406	Not Acceptable		
407	Proxy Authentication Required		
408	Request Timeout		
409	Conflict		
410	Gone	The requested resource is no longer available at the server and no forwarding address is known. This is the preferred way of letting the http client know that a resource is permanently no longer available.	

Example usage: Redirect gone /foo <http://www.example.com/bar> | |

41 1	Length Required		
41 2	Precondition Failed		
41 3	Request Entity Too Large		
41 4	Request-URI Too Long		
41 5	Unsupported Media Type		
41 6	Requested Range Not Satisfiable		
41 7	Expectation Failed		
42 2	Unprocessable Entity	Part of the http status code extensions for Distributed Authoring (WebDAV). Used to let the http client know that the server understands the content type of the request entity but was unable to process the contained instructions.	

Example usage: a request body contains syntactically correct XML but semantically erroneous XML instructions. [RFC 2518 Section 10](#) |

42 3	Locked	Part of the http status code extensions for Distributed Authoring (WebDAV). Means the source or destination resource of a method is locked.
---------	--------	--

Example usage: Alice is working on file foo.txt in a svn repository and locks the file at the end of her shift. In the mean time Bob was also working on foo.txt and tries to commit his changes through a WebDAV-enabled httpd, which results in a response code 423. [RFC 2518 Section 10](#) |

42 4	Failed Dependency	Part of the http status code extensions for Distributed Authoring (WebDAV). Means that the method could not be performed on the resource because the requested action depended on another action and that action failed.
---------	-------------------	---

Example usage: a PROPPATCH method is invoked with multiple changes to a resource, and one of these modifications fails. Since this method executes as a single atomic act, all changes must be able to be processed for the request to be successfully completed. The other changes depended on the single failed modification, and the httpd returns status code 424. [RFC 2518 Section 10](#) |

4 2 6	Upgrade Required	Part of the http status code extensions for Upgrading to TLS Within HTTP/1.1. Lets the client know that TLS is required for the client request to be completed. Will be followed by a Upgrade header specifying the required TLS version.	RFC 2817
	5xx - Server Error		
5 0 0	Internal Server Error	The server encountered an unexpected condition which prevented it from fulfilling the request. Basically means that "something" went wrong on the server. You will need to consult the errorlog for what exactly went wrong.	

Common causes are:

- * Erroneous CGI / PHP (Bad filesystem permissions, scripting errors, malformed or missing shebang line, bad line endings, script timeouts, etc)
- * Malformed directives in a .htaccess file
- * Hardware / OS problems

Check out the link to detailed errors for more detailed error messages. [Detailed errors](#) |

5 0 1	Not Implemented	The server does not support the functionality required to fulfill the request. This is occasionally seen in cases where a malicious client tries to execute commands through the httpd. Requests like "x05x01" arrive at the server, which is replied with a 501 status code.	
5 0 2	Bad Gateway	The server, while acting as a gateway or proxy, received an invalid response from the upstream server it accessed in attempting to fulfill the request. This can happen when a backend processor is misconfigured or encounters an error. Also the default behaviour of mod_proxy is to send the status code 502 if it receives syntactically invalid response header lines (i.e. containing no colon) from the origin server. A bug causing Bad Gateway errors was fixed in 2.2.4, so if you encounter this error using an old version, please upgrade to the most recent version.	Proxy BadHeader
5 0 3	Service Unavailable	The server is currently unable to handle the request due to a temporary overloading or maintenance of the server. Check the current load of the server using for example mod_status or your OS (for example: top, ps, apachetop). Could also be caused by misconfigured MPM settings.	
5 0 4	Gateway Timeout	The server, while acting as a gateway or proxy, did not receive a timely response from the upstream server specified by the URI (e.g. HTTP, FTP, LDAP) or some other auxiliary server (e.g. DNS) it needed to access in attempting to complete the request. You will need to check the status or logs on the backend side to find out what is causing the timeouts.	
5 0 5	HTTP Version Not Supported	The http version used in the request is not supported by the httpd. Note that the Apache HTTPD Server is fully HTTP/1.1 compatible since version 1.2 (released June 1997), so this error will not likely be encountered, unless a malformed version was specified in the request.	
5 0 6	Variant Also Negotiates	Part of the http status code extensions for Transparent Content Negotiation in HTTP. Indicates that the server has an internal configuration error: the chosen variant resource is configured to engage in transparent content negotiation itself, and is therefore not a proper end point in the negotiation process.	

Example usage: _  TODO: Documentation says this RFC is supported, but what config or type-map contents would result in this status code?_ [RFC 2295 Section 8](#) |

5 0 7	Insufficient Storage	Part of the http status code extensions for Distributed Authoring (WebDAV). Means the method could not be performed on the resource because the server is unable to store the representation needed to successfully complete the request. Also known to be used by other applications, such as Microsoft Exchange.	RFC 2518 Section 10
-------------	----------------------	---	-------------------------------------

5 1 0	Not Extended	Part of the http status code extensions for An HTTP Extension Framework. See link for details.	RFC 2774 Section 7
-------------	-----------------	--	--

To get more details on the complete meaning of these codes you should consult the [relevant standards](#), or more specifically [RFC 2616 Section 10](#).

HTTP status codes can be found in the Apache HTTPD logfiles, this example shows a request that results in a '200' response code:

```
127.0.0.1 - foo [21/Jul/2011:13:37:42 +0100] "GET /bar HTTP/1.1" 200 23674
```

Remember that, in the case of an error status code (4xx or 5xx), there will almost always be more details available in the Apache error log. See [DistrosDefaultLayout](#) if you don't know where that file is kept.