ВикипедиЯ

Список кодов состояния HTTP

Материал из Википедии — свободной энциклопедии

Код состояния HTTP (англ. *HTTP status code*) — часть первой строки ответа сервера при запросах по протоколу HTTP. Он представляет собой целое число из трёх десятичных цифр. Первая цифра указывает на ***класс состояния***. За кодом ответа обычно следует отделённая пробелом поясняющая фраза на английском языке, которая разъясняет человеку причину именно такого ответа. Примеры:

- 201 Created.
- 401 Unauthorized.
- 507 Insufficient Storage.

Клиент узнаёт по коду ответа о результатах его запроса и определяет, какие действия ему предпринимать дальше. Набор кодов состояния является стандартом, и они описаны в соответствующих документах RFC. Введение новых кодов должно производиться только после согласования с IETF. Тем не менее известно о двух используемых кодах, не упомянутых в RFC: `449 Retry With`. Также упоминается пояснительная фраза «Reply With»[1] в спецификации по *WebDAV* в *Microsoft Developer Network*, введённый *Microsoft* и `509 Bandwidth Limit Exceeded`, введённый в *cPanel*.

Клиент может не знать все коды состояния, но он обязан отреагировать в соответствии с классом кода. В настоящее время выделено пять классов кодов состояния.

Веб-сервер *Internet Information Services* в своих файлах журналов, кроме стандартных кодов состояния, использует подкоды, записывая их через точку после основного. При этом в ответах от сервера данный подкод не размещается — он нужен администратору сервера, чтобы тот мог более точно определять источники проблем.

HTTP

Постоянное соединение · HTTP pipelining · Сжатие · HTTPS · HTTP/2

Методы

OPTIONS · GET · HEAD · POST · PUT · DELETE · TRACE · CONNECT · PATCH

Заголовки

Cookie · ETag · Referer
HTTP location
Do Not Track
X-Forwarded-For

Коды состояния

1xx: Informational

2xx: Success

3xx: Redirection

4xx: Client Error (404 Not Found)

5xx: Server Error

Содержание

Обзорный список

Ниже представлен обзорный список всех описанных в данной статье кодов ответа:

- 1xx: Informational (информационные):
  - 100 Continue («продолжай»)[2][3];
  - 101 Switching Protocols («переключение протоколов»)[2][3];
  - 102 Processing («идёт обработка»).
- 2xx: Success (успешно):
  - 200 OK («хорошо»)[2][3];
  - 201 Created («создано»)[2][3][4];
  - 202 Accepted («принято»)[2][3];
  - 203 Non-Authoritative Information («информация не авторитетна»)[2][3];
  - 204 No Content («нет содержимого»)[2][3];
  - 205 Reset Content («сбросить содержимое»)[2][3];
  - 206 Partial Content («частичное содержимое»)[2][3];
  - 207 Multi-Status («многостатусный»)[5];
  - 208 Already Reported («уже сообщалось»)[6];
  - 226 IM Used («использовано IM»).
- 3xx: Redirection (перенаправление):
  - 300 Multiple Choices («множество выборов»)[2][7];
  - 301 Moved Permanently («перемещено навсегда»)[2][7];
  - 302 Moved Temporarily («перемещено временно»)[2][7];
  - 302 Found («найдено»)[7];
  - 303 See Other («смотреть другое»)[2][7];
  - 304 Not Modified («не изменялось»)[2][7];
  - 305 Use Proxy («использовать прокси»)[2][7];
  - 306 — *зарезервировано* (код использовался только в ранних спецификациях)[7];
  - 307 Temporary Redirect («временное перенаправление»)[7];
  - 308 Permanent Redirect («постоянное перенаправление»)[8].
- 4xx: Client Error (ошибка клиента):
  - 400 Bad Request («плохой, неверный запрос»)[2][3][4];
  - 401 Unauthorized («не авторизован (не представился)»)[2][3];
  - 402 Payment Required («необходима оплата»)[2][3];
  - 403 Forbidden («запрещено (не уполномочен)»)[2][3];
  - 404 Not Found («не найдено»)[2][3];
  - 405 Method Not Allowed («метод не поддерживается»)[2][3];
  - 406 Not Acceptable («неприемлемо»)[2][3];

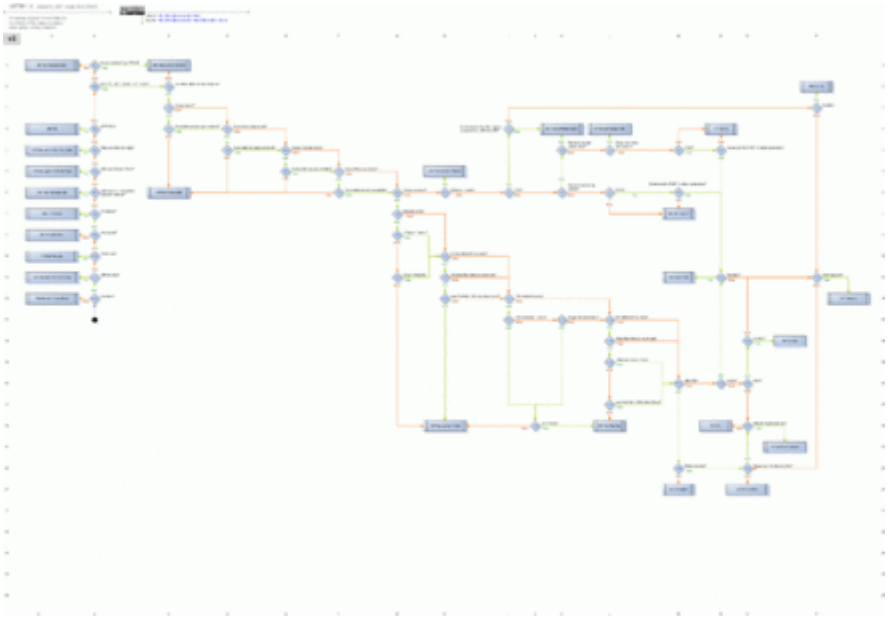

Диаграмма принятия веб-сервером решений на основе заголовков

Hits by Response Code		
Code 200 - OK	15.45%	7527
Code 206 - Partial Content	0.13%	64
Code 301 - Moved Permanently	0.03%	14
Code 302 - Found	0.18%	90
Code 304 - Not Modified	83.46%	40656
Code 403 - Forbidden	0.00%	1
Code 404 - Not Found	0.72%	353
Code 500 - Internal Server Error	0.02%	10

Статистика по кодам ответа, сгенерированная анализатором логов *Webalizer*

- 407 Proxy Authentication Required («необходима аутентификация прокси»)[2][3];
- 408 Request Timeout («истекло время ожидания»)[2][3];
- 409 Conflict («конфликт»)[2][3][4];
- 410 Gone («удалён»)[2][3];
- 411 Length Required («необходима длина»)[2][3];
- 412 Precondition Failed («условие ложно»)[2][3][9];
- 413 Payload Too Large («полезная нагрузка слишком велика»)[2][3];
- 414 URI Too Long («URI слишком длинный»)[2][3];
- 415 Unsupported Media Type («неподдерживаемый тип данных»)[2][3];
- 416 Range Not Satisfiable («диапазон не достижим»)[3];
- 417 Expectation Failed («ожидание не удалось»)[3];
- 418 I'm a teapot («я — чайник»);
- 419 Authentication Timeout (not in RFC 2616) («обычно ошибка проверки CSRF»);
- 421 Misdirected Request [10];
- 422 Unprocessable Entity («необрабатываемый экземпляр»);
- 423 Locked («заблокировано»);
- 424 Failed Dependency («невыполненная зависимость»);
- 426 Upgrade Required («необходимо обновление»);
- 428 Precondition Required («необходимо предусловие»)[11];
- 429 Too Many Requests («слишком много запросов»)[11];
- 431 Request Header Fields Too Large («поля заголовка запроса слишком большие»)[11];
- 449 Retry With («повторить с»)[1];
- 451 Unavailable For Legal Reasons («недоступно по юридическим причинам»)[12].
- 499 Client Closed Request (клиент закрыл соединение);

- 5xx: Server Error (ошибка сервера):
  - 500 Internal Server Error («внутренняя ошибка сервера»)[2][3];
  - 501 Not Implemented («не реализовано»)[2][3];
  - 502 Bad Gateway («плохой, ошибочный шлюз»)[2][3];
  - 503 Service Unavailable («сервис недоступен»)[2][3];
  - 504 Gateway Timeout («шлюз не отвечает»)[2][3];
  - 505 HTTP Version Not Supported («версия HTTP не поддерживается»)[2][3];
  - 506 Variant Also Negotiates («вариант тоже проводит согласование»)[13];
  - 507 Insufficient Storage («переполнение хранилища»);
  - 508 Loop Detected («обнаружено бесконечное перенаправление»)[14];
  - 509 Bandwidth Limit Exceeded («исчерпана пропускная ширина канала»);
  - 510 Not Extended («не расширено»);
  - 511 Network Authentication Required («требуется сетевая аутентификация»)[11];
  - 520 Unknown Error («неизвестная ошибка»)[15];
  - 521 Web Server Is Down («веб-сервер не работает»)[15];
  - 522 Connection Timed Out («соединение не отвечает»)[15];
  - 523 Origin Is Unreachable («источник недоступен»)[15];
  - 524 A Timeout Occurred («время ожидания истекло»)[15];
  - 525 SSL Handshake Failed («квитирование SSL не удалось»)[15];
  - 526 Invalid SSL Certificate («недействительный сертификат SSL»)[15].

Описание кодов

Информационные

В этот класс выделены коды, информирующие о процессе передачи. При работе через протокол версии 1.0 сообщения с такими кодами должны игнорироваться. В версии 1.1 клиент должен быть готов принять этот класс сообщений как обычный ответ, но серверу отправлять что-либо не нужно. Сами сообщения от сервера содержат только стартовую строку ответа и, если требуется, несколько специфичных для ответа полей заголовка. Прокси-сервера подобные сообщения должны отправлять дальше от сервера к клиенту.

- 100 Continue — сервер удовлетворён начальными сведениями о запросе, клиент может продолжать пересылать заголовки. Появился в HTTP/1.1.
- 101 Switching Protocols — сервер выполняет требование клиента и переключает протоколы в соответствии с указанием, данным в поле заголовка `Upgrade`. Сервер отправляет заголовок ответа `Upgrade`, указывая протокол, на который он переключился. Появился в HTTP/1.1.
- 102 Processing — запрос принят, но на его обработку понадобится длительное время. Используется сервером, чтобы клиент не разорвал соединение из-за превышения времени ожидания. Клиент при получении такого ответа должен сбросить таймер и дожидаться следующей команды в обычном режиме. Появился в *WebDAV*.

Успех

Сообщения данного класса информируют о случаях успешного принятия и обработки запроса клиента. В зависимости от статуса сервер может ещё передать заголовки и тело сообщения.

- 200 OK — успешный запрос. Если клиентом были запрошены какие-либо данные, то они находятся в заголовке и/или теле сообщения. Появился в HTTP/1.0.
- 201 Created — в результате успешного выполнения запроса был создан новый ресурс. Сервер может указать адреса (их может быть несколько) созданного ресурса в теле ответа, при этом предпочтительный адрес указывается в заголовке `Location`. Серверу рекомендуется указывать в теле ответа характеристики созданного ресурса и его адреса, формат тела ответа определяется заголовком `Content-Type`. При обработке запроса новый ресурс должен быть создан до отправки ответа клиенту, иначе следует использовать ответ с кодом `202`. Появился в HTTP/1.0.
- 202 Accepted — запрос был принят на обработку, но она не завершена. Клиенту не обязательно дожидаться окончательной передачи сообщения, так как может быть начат очень долгий процесс. Появился в HTTP/1.0.
- 203 Non-Authoritative Information — аналогично ответу `200`, но в этом случае передаваемая информация была взята не из первичного источника (резервной копии, другого сервера и т. д.) и поэтому может быть неактуальной. Появился в HTTP/1.1.
- 204 No Content — сервер успешно обработал запрос, но в ответе были переданы только заголовки без тела сообщения. Клиент не должен обновлять содержимое документа, но может применить к нему полученные метаданные. Появился в HTTP/1.0.
- 205 Reset Content — сервер обязывает клиента сбросить введённые пользователем данные. Тела сообщения сервер при этом не передаёт и документ обновлять не обязательно. Появился в HTTP/1.1.
- 206 Partial Content — сервер удачно выполнил частичный GET-запрос, возвратив только часть сообщения. В заголовке `Content-Range` сервер указывает байтовые диапазоны содержимого. Особое внимание при работе с подобными ответами следует уделить кэшированию. Появился в HTTP/1.1. (*подробнее…*)
- 207 Multi-Status — сервер передаёт результаты выполнения сразу нескольких независимых операций. Они помещаются в само тело сообщения в виде XML-документа с объектом `multistatus`. Не рекомендуется размещать в этом объекте статусы из серии `1xx` из-за бессмысленности и избыточности. Появился в *WebDAV*.
- 208 Already Reported — члены привязки DAV уже были перечислены в предыдущей части (multistatus) ответа и не включаются снова.
- 226 IM Used — заголовок `A-IM` от клиента был успешно принят и сервер возвращает содержимое с учётом указанных параметров. Введено в RFC 3229 для дополнения протокола HTTP поддержкой дельта-кодирования.

Перенаправление

Коды этого класса сообщают клиенту, что для успешного выполнения операции необходимо сделать другой запрос, как правило, по другому URI. Из данного класса пять кодов `301`, `302`, `303`, `305` и `307` относятся непосредственно к перенаправлениям. Адрес, по которому клиенту следует произвести запрос, сервер указывает в заголовке `Location`. При этом допускается использование фрагментов в целевом URI.

По последним стандартам клиент может производить перенаправление без запроса пользователя только если второй ресурс будет запрашиваться методом `GET` или `HEAD`[7]. В предыдущих спецификациях говорилось, что для избежания круговых переходов пользователя следует спрашивать после 5-го подряд перенаправления[16]. При всех перенаправлениях, если метод запроса был не `HEAD`, то в тело ответа следует включить короткое гипертекстовое сообщение с целевым адресом, чтобы в случае ошибки пользователь смог сам произвести переход.

Разработчики HTTP отмечают, что многие клиенты при перенаправлениях с кодами `301` и `302` ошибочно применяют метод `GET` ко второму ресурсу, несмотря на то, что к первому запрос был с иным методом (чаще всего PUT)[17]. Чтобы избежать недоразумений, в версии HTTP/1.1 были введены коды `303` и `307` и их рекомендовано использовать вместо `302`. Изменять метод нужно только если сервер ответил `303`. В остальных случаях следующий запрос производить с исходным методом.

Поведение клиентов при различных перенаправлениях описано в таблице:

Статус ответа	Кэширование	Если метод не GET или HEAD
`301 Moved Permanently`	Можно как обычно.	Спросить у пользователя подтверждения и запросить другой ресурс исходным методом.
`307 Temporary Redirect`	Можно только если указан заголовок `Cache-Control` или `Expires`.	
`302 Found` (HTTP/1.1) `302 Moved Temporarily` (HTTP/1.0)		
`303 See Other`	Нельзя.	Перейти автоматически, но уже методом `GET`.

- 300 Multiple Choices — по указанному URI существует несколько вариантов предоставления ресурса по типу MIME, по языку или по другим характеристикам. Сервер передаёт с сообщением список альтернатив, давая возможность сделать выбор клиенту автоматически или пользователю. Появился в HTTP/1.0.
- 301 Moved Permanently — запрошенный документ был окончательно перенесен на новый URI, указанный в поле `Location` заголовка. Некоторые клиенты некорректно ведут себя при обработке данного кода. Появился в HTTP/1.0.
- 302 Found, 302 Moved Temporarily — запрошенный документ временно доступен по другому URI, указанному в заголовке в поле `Location`. Этот код может быть использован, например, при управляемом сервером согласовании содержимого. Некоторые[*какие?*] клиенты некорректно ведут себя при обработке данного кода. Введено в HTTP/1.0.
- 303 See Other — документ по запрошенному URI нужно запросить по адресу в поле `Location` заголовка с использованием метода `GET` несмотря даже на то, что первый запрашивался иным методом. Этот код был введён вместе с кодом `307` для избежания неоднозначности, чтобы сервер был уверен, что следующий ресурс будет запрошен методом `GET`. Например, на веб-странице есть поле ввода текста для быстрого перехода и поиска. После ввода данных браузер делает запрос методом `POST`, включая в тело сообщения введённый текст. Если обнаружен документ с введённым названием, то сервер отвечает кодом `303`, указав в заголовке `Location` его постоянный адрес. Тогда браузер гарантировано его запросит методом `GET` для получения содержимого. В противном случае сервер просто вернёт клиенту страницу с результатами поиска. Введено в HTTP/1.1.
- 304 Not Modified — сервер возвращает такой код, если клиент запросил документ методом `GET`, использовал заголовок `If-Modified-Since` или `If-None-Match` и документ не изменился с указанного момента. При этом сообщение сервера не должно содержать тела. Появился в HTTP/1.0.
- 305 Use Proxy — запрос к запрашиваемому ресурсу должен осуществляться через прокси-сервер, URI которого указан в поле `Location` заголовка. Данный код ответа могут использовать только исходные HTTP-сервера (не прокси). Введено в HTTP/1.1.
- 306 (зарезервировано) — использовавшийся раньше код ответа, в настоящий момент зарезервирован. Упомянут в RFC 2616 (обновление HTTP/1.1).
- 307 Temporary Redirect — запрашиваемый ресурс на короткое время доступен по другому URI, указанный в поле `Location` заголовка. Метод запроса (GET/POST) менять не разрешается. Например, POST запрос должен быть отправлен по новому URI тем же методом POST. Этот код был введён вместе с 303 вместо 302-го для избежания неоднозначности. Введено в RFC 2616 (обновление HTTP/1.1).
- 308 Permanent Redirect — запрашиваемый ресурс был окончательно перенесен на новый URI, указанный в поле `Location` заголовка. Метод запроса (GET/POST) менять не разрешается. Например, POST запрос должен быть отправлен по новому URI тем же методом POST. Этот код был введён вместо 301-го для избежания неоднозначности. Введено в RFC 7238 (RFC 7538).

Ошибка клиента

Класс кодов `4xx` предназначен для указания ошибок со стороны клиента. При использовании всех методов, кроме `HEAD`, сервер должен вернуть в теле сообщения гипертекстовое пояснение для пользователя.

- 400 Bad Request — сервер обнаружил в запросе клиента синтаксическую ошибку. Появился в HTTP/1.0.
- 401 Unauthorized — для доступа к запрашиваемому ресурсу требуется аутентификация. В заголовке ответ должен содержать поле `WWW-Authenticate` с перечнем условий аутентификации. Иными словами, для доступа к запрашиваемому ресурсу клиент должен представиться, послав запрос, включив при этом в заголовок сообщения поле `Authorization` с требуемыми для аутентификации данными.
- 402 Payment Required — предполагается использовать в будущем. В настоящий момент не используется. Этот код предусмотрен для платных пользовательских сервисов, а не для хостинговых компаний. Имеется в виду, что эта ошибка не будет выдана хостинговым провайдером в случае просроченной оплаты его услуг. Зарезервирован, начиная с HTTP/1.1.
- 403 Forbidden[18] — сервер понял запрос, но он отказывается его выполнять из-за ограничений в доступе для клиента к указанному ресурсу. Иными словами, клиент не уполномочен совершать операции с запрошенным ресурсом. Если для доступа к ресурсу требуется аутентификация средствами HTTP, то сервер вернёт ответ `401`, или `407` при использовании прокси. В противном случае ограничения были заданы администратором сервера или разработчиком веб-приложения и могут быть любыми в зависимости от возможностей используемого программного обеспечения. В любом случае клиенту следует сообщить причины отказа в обработке запроса. Наиболее вероятными причинами ограничения может послужить попытка доступа к системным ресурсам веб-сервера (например, файлам `.htaccess` или `.htpasswd`) или к файлам, доступ к которым был закрыт с помощью конфигурационных файлов, требование аутентификации не средствами HTTP, например, для доступа к системе управления содержимым или разделу для зарегистрированных пользователей либо сервер не удовлетворён IP-адресом клиента, например, при блокировках. Появился в HTTP/1.0.

Сервер вернул ошибку 403 при попытке просмотра директории «cgi-bin», доступ к которой был запрещён.

- 404 Not Found[19] — самая распространённая ошибка при пользовании Интернетом, основная причина — ошибка в написании адреса Web-страницы. Сервер понял запрос, но не нашёл соответствующего ресурса по указанному URL. Если серверу известно, что по этому адресу был документ, то ему желательно использовать код `410`. Ответ `404` может использоваться вместо `403`, если требуется тщательно скрыть от посторонних глаз определённые ресурсы. Появился в HTTP/1.0.
- 405 Method Not Allowed — указанный клиентом метод нельзя применить к текущему ресурсу. В ответе сервер должен указать доступные методы в заголовке `Allow`, разделив их запятой. Эту ошибку сервер должен возвращать, если метод ему известен, но он не применим именно к указанному в запросе ресурсу, если же указанный метод не применим на всём сервере, то клиенту нужно вернуть код `501` (Not Implemented). Появился в HTTP/1.1.
- 406 Not Acceptable — запрошенный URI не может удовлетворить переданным в заголовке характеристикам. Если метод был не `HEAD`, то сервер должен вернуть список допустимых характеристик для данного ресурса. Появился в HTTP/1.1.
- 407 Proxy Authentication Required — ответ аналогичен коду `401` за исключением того, что аутентификация производится для прокси-сервера. Механизм аналогичен идентификации на исходном сервере. Появился в HTTP/1.1.
- 408 Request Timeout — время ожидания сервером передачи от клиента истекло. Клиент может повторить аналогичный предыдущему запрос в любое время. Например, такая ситуация может возникнуть при загрузке на сервер объёмного файла методом `POST` или `PUT`. В какой-то момент передачи источник данных перестал отвечать, например, из-за повреждения компакт-диска или потери связи с другим компьютером в локальной сети. Пока клиент ничего не передаёт, ожидая от него ответа, соединение с сервером держится. Через некоторое время сервер может закрыть соединение со своей стороны, чтобы дать возможность другим клиентам сделать запрос. Этот ответ не возвращается, когда клиент принудительно остановил передачу по команде пользователя или соединение прервалось по каким-то иным причинам, так как ответ уже послать невозможно. Появился в HTTP/1.1.
- 409 Conflict — запрос не может быть выполнен из-за конфликтного обращения к ресурсу. Такое возможно, например, когда два клиента пытаются изменить ресурс с помощью метода PUT.Появился в HTTP/1.1.
- 410 Gone — такой ответ сервер посылает, если ресурс раньше был по указанному URL, но был удалён и теперь недоступен. Серверу в этом случае неизвестно и местоположение альтернативного документа (например копии). Если у сервера есть подозрение, что документ в ближайшее время может быть восстановлен, то лучше клиенту передать код `404`. Появился в HTTP/1.1.

- 411 Length Required — для указанного ресурса клиент должен указать `Content-Length` в заголовке запроса. Без указания этого поля не стоит делать повторную попытку запроса к серверу по данному URI. Такой ответ естественен для запросов типа `POST` и `PUT`. Например, если по указанному URI производится загрузка файлов, а на сервере стоит ограничение на их объём. Тогда разумней будет проверить в самом начале заголовок `Content-Length` и сразу отказать в загрузке, чем провоцировать бессмысленную нагрузку, разрывая соединение, когда клиент действительно пришлёт слишком объёмное сообщение. Появился в HTTP/1.1.
- 412 Precondition Failed — возвращается, если ни одно из условных полей заголовка (If-Match и др., см. RFC 7232) запроса не было выполнено. Появился в HTTP/1.1.
- 413 Payload Too Large — возвращается в случае, если сервер отказывается обработать запрос по причине слишком большого размера тела запроса. Сервер может закрыть соединение, чтобы прекратить дальнейшую передачу запроса. Если проблема временная, то рекомендуется в ответ сервера включить заголовок `Retry-After` с указанием времени, по истечении которого можно повторить аналогичный запрос. Появился в HTTP/1.1. Ранее назывался «Request Entity Too Large».
- 414 URI Too Long — сервер не может обработать запрос из-за слишком длинного указанного URI. Такую ошибку можно спровоцировать, например, когда клиент пытается передать длинные параметры через метод `GET`, а не `POST`. Появился в HTTP/1.1. Ранее назывался «Request-URI Too Long».
- 415 Unsupported Media Type — по каким-то причинам сервер отказывается работать с указанным типом данных при данном методе. Появился в HTTP/1.1.
- 416 Range Not Satisfiable — в поле `Range` заголовка запроса был указан диапазон за пределами ресурса и отсутствует поле `If-Range`. Если клиент передал байтовый диапазон, то сервер может вернуть реальный размер в поле `Content-Range` заголовка. Данный ответ не следует использовать при передаче типа `multipart/byteranges`. Введено в RFC 2616 (обновление HTTP/1.1). Ранее назывался «Requested Range Not Satisfiable».
- 417 Expectation Failed — по каким-то причинам сервер не может удовлетворить значению поля `Expect` заголовка запроса. Введено в RFC 2616 (обновление HTTP/1.1).
- 418 I'm a teapot — Этот код был введен в 1998 году как одна из традиционных первоапрельских шуток IETF в RFC 2324, Hyper Text Coffee Pot Control Protocol. Не ожидается, что данный код будет поддерживаться реальными серверами[20].
- 419 Authentication Timeout (not in RFC 2616) — Этого кода нет в RFC 2616, используется в качестве альтернативы коду 401, которые прошли проверку подлинности, но лишены доступа к определенным ресурсам сервера. Обычно код отдается, если CSRF токен устарел или оказался некорректным.
- 421 Misdirected Request — запрос был перенаправлен на сервер, не способный дать ответ.
- 422 Unprocessable Entity — сервер успешно принял запрос, может работать с указанным видом данных (например, в теле запроса находится XML-документ, имеющий верный синтаксис), однако имеется какая-то логическая ошибка, из-за которой невозможно произвести операцию над ресурсом. Введено в *WebDAV*.
- 423 Locked — целевой ресурс из запроса заблокирован от применения к нему указанного метода. Введено в WebDAV.
- 424 Failed Dependency — реализация текущего запроса может зависеть от успешности выполнения другой операции. Если она не выполнена и из-за этого нельзя выполнить текущий запрос, то сервер вернёт этот код. Введено в *WebDAV*.
- 426 Upgrade Required — сервер указывает клиенту на необходимость обновить протокол. Заголовок ответа должен содержать правильно сформированные поля `Upgrade` и `Connection`. Введено в RFC 2817 для возможности перехода к TLS посредством HTTP.
- 428 Precondition Required — сервер указывает клиенту на необходимость использования в запросе заголовков условий, наподобие `If-Match`. Введено в черновике стандарта RFC 6585.
- 429 Too Many Requests — клиент попытался отправить слишком много запросов за короткое время, что может указывать, например, на попытку DDoS-атаки. Может сопровождаться заголовком Retry-After, указывающим, через какое время можно повторить запрос. Введено в черновике стандарта RFC 6585.
- 431 Request Header Fields Too Large — Превышена допустимая длина заголовков. Сервер не обязан отвечать этим кодом, вместо этого он может просто сбросить соединение. Введено в черновике стандарта RFC 6585.
- 434 Requested host unavailable — Запрашиваемый адрес недоступен.
- 449 Retry With — возвращается сервером, если для обработки запроса от клиента поступило недостаточно информации. При этом в заголовок ответа помещается поле `Ms-Echo-Request`. Введено корпорацией *Microsoft* для *WebDAV*. В настоящий момент как минимум используется программой *Microsoft Money*.
- 451 Unavailable For Legal Reasons — доступ к ресурсу закрыт по юридическим причинам, например, по требованию органов государственной власти или по требованию правообладателя в случае нарушения авторских прав. Введено в черновике IETF за авторством Google[12], при этом код ошибки является отсылкой к роману Рэя Брэдбери «451 градус по Фаренгейту». Был добавлен в стандарт 21 декабря 2015[21].
- 499 Client Closed Request - нестандартный код, предложенный и используемый nginx для случаев когда клиент закрыл соединение, пока nginx обрабатывал запрос

Ошибка сервера

Коды `5xx` выделены под случаи неудачного выполнения операции по вине сервера. Для всех ситуаций, кроме использования метода `HEAD`, сервер должен включать в тело сообщения объяснение, которое клиент отобразит пользователю.

Пример ошибки 502 Bad Gateway

- 500 Internal Server Error[22] — любая внутренняя ошибка сервера, которая не входит в рамки остальных ошибок класса. Появился в HTTP/1.0.
- 501 Not Implemented — сервер не поддерживает возможностей, необходимых для обработки запроса. Типичный ответ для случаев, когда сервер не понимает указанный в запросе метод. Если же метод серверу известен, но он не применим к данному ресурсу, то нужно вернуть ответ `405`. Появился в HTTP/1.0.
- 502 Bad Gateway — сервер, выступая в роли шлюза или прокси-сервера, получил недействительное ответное сообщение от вышестоящего сервера. Появился в HTTP/1.0.
- 503 Service Unavailable — сервер временно не имеет возможности обрабатывать запросы по техническим причинам (обслуживание, перегрузка и прочее). В поле `Retry-After` заголовка сервер может указать время, через которое клиенту рекомендуется повторить запрос. Хотя во время перегрузки очевидным кажется сразу разрывать соединение, эффективней может оказаться установка большого значения поля `Retry-After` для уменьшения частоты избыточных запросов. Появился в HTTP/1.0.
- 504 Gateway Timeout — сервер в роли шлюза или прокси-сервера не дождался ответа от вышестоящего сервера для завершения текущего запроса. Появился в HTTP/1.1.
- 505 HTTP Version Not Supported — сервер не поддерживает или отказывается поддерживать указанную в запросе версию протокола HTTP. Появился в HTTP/1.1.
- 506 Variant Also Negotiates — в результате ошибочной конфигурации выбранный вариант указывает сам на себя, из-за чего процесс связывания прерывается. Экспериментальное. Введено в RFC 2295 для дополнения протокола HTTP технологией *Transparent Content Negotiation*.
- 507 Insufficient Storage — не хватает места для выполнения текущего запроса. Проблема может быть временной. Введено в *WebDAV*.
- 509 Bandwidth Limit Exceeded — используется при превышении веб-площадкой отведённого ей ограничения на потребление трафика. В данном случае владельцу площадки следует обратиться к своему хостинг-провайдеру. В настоящий момент данный код не описан ни в одном RFC и используется только модулем «bw/limited», входящим в панель управления хостингом *cPanel*, где и был введён.
- 510 Not Extended — на сервере отсутствует расширение, которое желает использовать клиент. Сервер может дополнительно передать информацию о доступных ему расширениях. Введено в RFC 2774 для дополнения протокола HTTP поддержкой расширений.
- 511 Network Authentication Required — этот ответ посылается не сервером, которому был предназначен запрос, а сервером-посредником — например, сервером провайдера — в случае, если клиент должен сначала авторизоваться в сети, например, ввести пароль для платной точки доступа к Интернету. Предполагается, что в теле ответа будет возвращена Web-форма авторизации или перенаправление на неё. Введено в черновике стандарта RFC 6585.
- 520 Unknown Error, возникает когда сервер CDN не смог обработать ошибку веб-сервера; нестандартный код CloudFlare.
- 521 Web Server Is Down, возникает когда подключения CDN отклоняются веб-сервером; нестандартный код CloudFlare.
- 522 Connection Timed Out, возникает когда CDN не удалось подключиться к веб-серверу; нестандартный код CloudFlare.
- 523 Origin Is Unreachable, возникает когда веб-сервер недостижим; нестандартный код CloudFlare.
- 524 A Timeout Occurred, возникает при истечении таймаута подключения между сервером CDN и веб-сервером; нестандартный код CloudFlare.
- 525 SSL Handshake Failed, возникает при ошибке рукопожатия SSL между сервером CDN и веб-сервером; нестандартный код CloudFlare.
- 526 Invalid SSL Certificate, возникает когда не удаётся подтвердить сертификат шифрования веб-сервера; нестандартный код CloudFlare.

См. также

- Список заголовков HTTP

- Код ответа

Примечания

1. *2.2.6 449 «Retry With Status Code»* // Web Distributed Authoring and Versioning (WebDAV) Protocol: Client Extensions. (http://msdn.microsoft.com/en-us/library/dd891478(PROT.10).aspx) на сайте MSDN
2. «6.1.1 Status Code and Reason Phrase (http://tools.ietf.org/html/rfc2068#section-6.1.1)» в RFC 2068
3. RFC 2616 (http://tools.ietf.org/html/rfc2616#section-10.3)
4. IETF Draft *WebDAV Advanced Collections Protocol* — S.4.2.5 (http://tools.ietf.org/html/draft-ietf-webdav-collection-protocol-04#section-4.2.5)
5. IETF Draft *WebDAV Advanced Collections Protocol* — S.10 (http://tools.ietf.org/html/draft-ietf-webdav-collection-protocol-04#section-9.13)
6. rfc5842 (https://tools.ietf.org/html/rfc5842).
7. RFC 2616 «10.3 Redirection 3xx» (стр. 61) (http://tools.ietf.org/html/rfc2616#section-10.3)
8. rfc7538 (https://tools.ietf.org/html/rfc7538).
9. IETF Draft *WebDAV Advanced Collections Protocol* — S.4.3.1.1 (http://tools.ietf.org/html/draft-ietf-webdav-collection-protocol-04#section-4.3.1)
10. rfc7540 (https://tools.ietf.org/html/rfc7540).
11. RFC 6585
12. IETF Draft *A New HTTP Status Code to Report Legal Obstacles* (http://tools.ietf.org/html/draft-tbray-http-legally-restricted-status-02)
13. RFC 2295 *Transparent Content Negotiation in HTTP* — S.8.1 (http://tools.ietf.org/html/rfc2295#section-8.1)
14. IETF Draft *WebDAV Advanced Collections Protocol* — S.7.1 (http://tools.ietf.org/html/draft-ietf-webdav-collection-protocol-04#section-7.1)
15. Error Pages – CloudFlare Support (https://support.cloudflare.com/hc/en-us/sections/200820298-Error-Pages)
16. RFC 2068 «10.3 Redirection 3xx» (стр. 56) (http://tools.ietf.org/html/rfc2068#section-10.3).
17. RFC 2616, раздел «10.3.3 302 Found», страница 63 (http://tools.ietf.org/html/rfc2616#page-63).
18. Что означает 403 Forbidden? (http://www.pageranker.ru/articles/troubleshooting/167--403-forbidden.html).
19. Причины появления ошибки 404 Not Found (http://www.pageranker.ru/articles/troubleshooting/168--404-not-found.html).
20. RFC 2324 — Hyper Text Coffee Pot Control Protocol (HTCPCP/1.0)
21. draft-ietf-httpbis-legally-restricted-status-04 (https://datatracker.ietf.org/doc/draft-ietf-httpbis-legally-restricted-status/?include_text=1). datatracker.ietf.org. Дата обращения 22 декабря 2015.
22. Описание ошибки 500 Internal Server Error (http://www.pageranker.ru/articles/troubleshooting/941-chto-oznachaet-oshibka-500-internal-server-error.html).

Ссылки

Основные документы по протоколу HTTP (по убыванию даты публикации)

- Hypertext Transfer Protocol (HTTP) Status Code Registry (http://www.iana.org/assignments/http-status-codes) (англ.). IANA (17 октября 2007). — реестр кодов состояния HTTP. Дата обращения 30 июля 2009. Архивировано (https://www.webcitation.org/65WVINQ6v?url=http://www.iana.org/assignments/http-status-codes/http-status-codes.xml#) 17 февраля 2012 года.
- RFC 2616 Draft standard «Hypertext Transfer Protocol — HTTP/1.1 (http://tools.ietf.org/html/rfc2068)» (англ.) (с англ. — «Протокол передачи гипертекста — HTTP/1.1»); IETF, июнь 1999; *Fielding Roy (UC Irvine), Gettys Jim (Compaq/W3C), Mogul J. (Compaq), Frystyk Henrik (MIT/W3C), Masinter L. (Xerox), Leach P. (Microsoft), Berners-Lee Tim (W3C/MIT)* — обновление протокола HTTP версии 1.1.
- RFC 2068 Proposed standard «Hypertext Transfer Protocol — HTTP/1.1 (http://tools.ietf.org/html/rfc2068)» (англ.) (с англ. — «Протокол передачи гипертекста — HTTP/1.1»); IETF, январь 1997; *Fielding Roy (UC Irvine), Gettys Jim (DEC), Mogul J. (DEC), Frystyk Henrik (MIT/LCS), Berners-Lee Tim (MIT/LCS)* — ранняя спецификация по HTTP версии 1.1.
- RFC 1945 Informational «Hypertext Transfer Protocol — HTTP/1.0 (http://tools.ietf.org/html/rfc1945)» (англ.) (с англ. — «Протокол передачи гипертекста — HTTP/1.0»); IETF, май 1996; *Berners-Lee Tim (MIT/LCS), Fielding Roy (UC Irvine), Frystyk Henrik (MIT/LCS)* — самая первая спецификация по протоколу HTTP. Так же включает в себя описание HTTP/0.9.

Документы по расширениям и обновлениям протокола HTTP (по убыванию даты публикации)

- RFC 4918 Proposed Standard «HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV) (http://tools.ietf.org/html/rfc4918)» (англ.) (с англ. — «Расширения HTTP для распределённой авторской работы и управления версиями через веб (WebDAV)»); IETF, июнь 2007; *Dusseault Ed. L. (CommerceNet)* — поздняя спецификация по протоколу WebDAV, заместившая RFC 2518.
- RFC 3229 Proposed standard «Delta encoding in HTTP (http://tools.ietf.org/html/rfc3229)» (англ.) (с англ. — «Дельта-кодирование в HTTP»); IETF, январь 2002; *Mogul J. (Compaq WRL), Krishnamurthy B. (AT&T), Douglis F. (AT&T), Feldmann A. (Univ. of Saarbrücken), Goland Y. (Marimba), van Hoff A. (Marimba), Hellerstein D. (ERS/USDA)*.
- RFC 2817 Proposed Standard «Upgrading to TLS Within HTTP/1.1 (http://tools.ietf.org/html/rfc2817)» (англ.) (с англ. — «Обновление к TLS совместно с HTTP/1.1»); IETF, май 2000; *Khare Rohit (4K Associates/UC Irvine), Lawrence S. (Agranat Systems, Inc.)* — обновление к RFC 2616 для описания работы HTTP и TLS.
- RFC 2774 Experimental «An HTTP Extension Framework (http://tools.ietf.org/html/rfc2774)» (англ.) (с англ. — «Каркас расширений HTTP»); IETF, февраль 2000; *Nielsen H. (Microsoft), Leach P. (Microsoft), Lawrence S. (Agranat Systems)*.
- Internet Draft «WebDAV Advanced Collections Protocol (http://tools.ietf.org/html/draft-ietf-webdav-collection-protocol-04)» (с англ. — «Протокол продвинутых коллекций WebDAV»); IETF, 18 июня 1999; *Slein J. (Xerox), Whitehead Jr. E. J. (UC Irvine), Davis J. (CourseNet), Clemm G. (Rational), Fay C. (FileNet), Crawford J. (IBM), Chihaya T. (DataChannel)* — управление коллекциями в WebDAV; просрочился 18 декабря 1999 года.
- RFC 2518 Proposed Standard «HTTP Extensions for Distributed Authoring — WEBDAV (http://tools.ietf.org/html/rfc2518)» (англ.) (с англ. — «Расширения HTTP для распределённой авторской работы — WEBDAV»); IETF, февраль 1999; *Goland Y. (Microsoft), Whitehead E. (UC Irvine), Faizi A. (Netscape), Carter S. (Novell), Jensen D. (Novell)* — первая спецификация по протоколу WebDAV (замещена RFC 4918).
- RFC 2295 Experimental «Transparent Content Negotiation in HTTP (http://tools.ietf.org/html/rfc2295)» (англ.) (с англ. — «Прозрачное согласование содержимого в HTTP»); IETF, март 1998; *Holtman K. (TUE), Mutz A. (Hewlett-Packard)*.

Дополнительные материалы

- Web Distributed Authoring and Versioning (WebDAV) Protocol: Client Extensions (http://msdn.microsoft.com/en-us/library/cc250046%28PROT.13%29.aspx) (англ.). Microsoft (14 марта 2007). — описание поддержки клиентских расширений в протоколе WebDAV. Дата обращения 30 июля 2009. Архивировано (https://www.webcitation.org/65WVloxS1?url=http://msdn.microsoft.com/en-us/library/cc250046(PROT.13).aspx#) 17 февраля 2012 года.
- RFC 2324 Informational «Hyper Text Coffee Pot Control Protocol (HTCPCP/1.0) (http://tools.ietf.org/html/rfc2774)» (англ.) (с англ. — «Гипертекстовый протокол управления кофеваркой (HTCPCP/1.0)»); IETF, 1 апреля 1998; *Masinter L.*.
- KB 318380 Коды состояния служб IIS (http://support.microsoft.com/kb/318380/). Microsoft (4 декабря 2007). — список расширенных кодов состояния для протоколов HTTP и FTP. Дата обращения 16 января 2010. Архивировано (https://www.webcitation.org/65WVmPE8h?url=http://support.microsoft.com/kb/318380/#) 17 февраля 2012 года.
- Справочник по кодам статуса HTTP (http://yandex.ru/support/webmaster/error-dictionary/http-codes.html). Яндекс. — обработка кодов состояния HTTP роботами Яндекса. Дата обращения 2 мая 2018.
