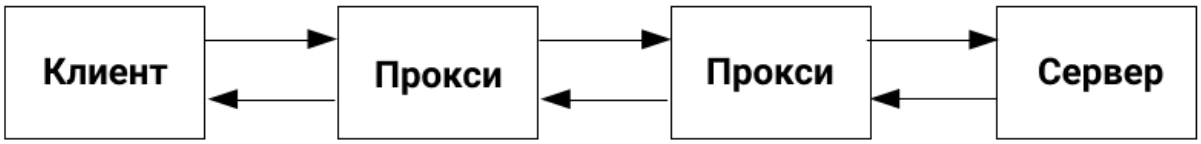
HTTP

Протокол **HTTP** предназначен для передачи содержимого в Интернете. HTTP — это простой протокол, который использует для передачи содержимого надежные службы протокола TCP. **HTTPS** — это безопасная версия протокола HTTP, которая реализует протокол HTTP с использованием протокола TLS для защиты базового TCP-подключения. HTTP основан на тексте — сообщения между клиентом и сервером по сути представляют собой фрагменты текста, хотя в теле сообщения могут быть другие элементы: видео, фото, аудио и т.д. Портом по умолчанию для HTTP является порт **80** или **8080**, но могут использоваться и другие порты. Для HTTPS порт **443**.

Типичные **расширения файлов**, используемые с протоколом HTTP:

- **HTM (или HTML):** HTML-файлы (HTML);
- **TXT**: открытый текст ASCII;
- **GIF**: двоичное изображение GIF;
- **XBM**: двоичное изображение Xbitmap.

Каждый отдельный запрос отправляется на сервер, который обрабатывает его и предоставляет ответ. Между клиентом и сервером существует множество объектов, которые называются прокси-серверами.

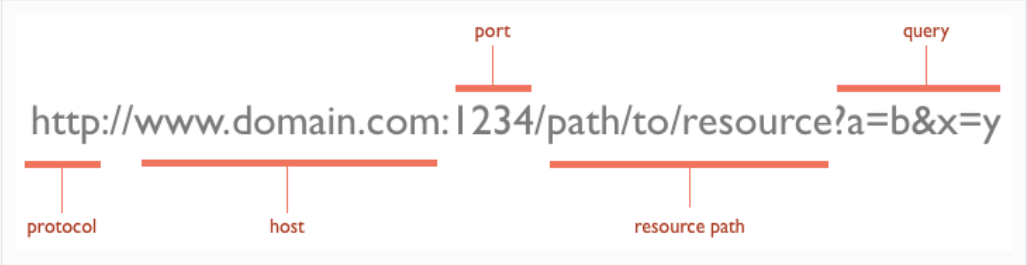


Схематичное изображение работы HTTP-протокола

Веб-разработчики могут использовать прокси для следующих целей: кэширование, аутентификация, логирование, веб-фильтрация, балансировка нагрузки.

Как работает HTTP-протокол

Шаг первый: направляем URL в браузер.



Шаг второй: браузер ищет нужный IP-адрес. Браузер использует преобразователь DNS для сопоставления домена с IP-адресом.

Шаг третий: браузер посылает HTTP-запрос. HTTP-запрос может состоять всего из двух строк текста, например:

```
GET/index.html HTTP/1.1
Host: www.example.com
```

Шаг четвертый: сервер отправляет HTTP-ответ. Например:

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Length: 208
```

Шаг пятый: отображается нужная веб-страница

Структура протокола

Структура протокола определяет, что каждое HTTP-сообщение состоит из трёх частей (рис. 1), которые передаются в следующем порядке:

- **Стартовая строка** (англ. Starting line) — определяет тип сообщения; Стартовые строки различаются для запроса и ответа. **Строка запроса** выглядит так:

  Метод URI HTTP/Версия протокола

  Пример запроса:

```
GET /web-programming/index.html HTTP/1.1
```

Стартовая **строка ответа** сервера имеет следующий формат:

```
HTTP/Версия КодСостояния [Пояснение]
```

Например, на предыдущий наш запрос клиентом данной страницы сервер ответил строкой:

```
HTTP/1.1 200 Ok
```

- **Заголовки** (англ. Headers) — это строка в HTTP-сообщении, содержащая разделённую двоеточием пару вида «параметр-значение». Например:

```
Server: Apache/2.2.3 (CentOS)
Last-Modified: Wed, 09 Feb 2011 17:13:15 GMT
Content-Type: text/html; charset=UTF-8
Accept-Ranges: bytes
Date: Thu, 03 Mar 2011 04:04:36 GMT
Content-Length: 2945
Age: 51
X-Cache: HIT from proxy.omgtu
Via: 1.0 proxy.omgtu (squid/3.1.8)
Connection: keep-alive
```

Заголовок	Группа	Краткое описание
Allow	Entity	Список методов, применимых к запрашиваемому ресурсу.
Content-Encoding	Entity	Применяется при необходимости перекодировки содержимого (например, gzip/deflated).
Content-Language	Entity	Локализация содержимого (язык(и))
Content-Length	Entity	Размер тела сообщения (в октетах)
Content-Range	Entity	Диапазон (используется для поддержания многопоточной загрузки или дозагрузки)
Content-Type	Entity	Указывает тип содержимого (mime-type, например text/html).Часто включает указание на таблицу символов локали (charset)
Expires	Entity	Дата/время, после которой ресурс считается устаревшим. Используется прокси-серверами
Last-Modified	Entity	Дата/время последней модификации сущности
Cache-Control	General	Определяет директивы управления механизмами кэширования. Для прокси-серверов.
Connection	General	Задает параметры, требуемые для конкретного соединения.
Date	General	Дата и время формирования сообщения
Pragma	General	Используется для специальных указаний, которые могут (опционально) применяется к любому получателю по всей цепочке запросов/ответов (например, pragma: no-cache).
Transfer-Encoding	General	Задает тип преобразования, применимого к телу сообщения. В отличие от Content-Encoding этот заголовок распространяется на все сообщение, а не только на сущность.
Via	General	Используется шлюзами и прокси для отображения промежуточных протоколов и узлов между клиентом и веб-сервером.
Warning	General	Дополнительная информация о текущем статусе, которая не может быть представлена в сообщении.
Accept	Request	Определяет применимые типы данных, ожидаемых в ответе.
Accept-Charset	Request	Определяет кодировку символов (charset) для данных, ожидаемых в ответе.

Заголовок	Группа	Краткое описание
Accept-Encoding	Request	Определяет применимые форматы кодирования/декодирования содержимого (напр, gzip)
Accept-Language	Request	Применимые языки. Используется для согласования передачи.
Authorization	Request	Учетные данные клиента, запрашивающего ресурс.
From	Request	Электронный адрес отправителя
Host	Request	Имя/сетевой адрес [и порт] сервера. Если порт не указан, используется 80.
If-Modified-Since	Request	Используется для выполнения условных методов (Если-Изменился...). Если запрашиваемый ресурс изменился, то он передается с сервера, иначе - из кэша.
Max-Forwards	Request	Представляет механиз ограничения количества шлюзов и прокси при использовании методов TRACE и OPTIONS.
Proxy-Authorization	Request	Используется при запросах, проходящих через прокси, требующие авторизации
Referer	Request	Адрес, с которого выполняется запрос. Этот заголовок отсутствует, если переход выполняется из адресной строки или, например, по ссылке из js-скрипта.
User-Agent	Request	Информация о пользовательском агенте (клиенте)
Location	Response	Адрес перенаправления
Proxy-Authenticate	Response	Сообщение о статусе с кодом 407.
Server	Response	Информация о программном обеспечении сервера, отвечающего на запрос (это может быть как веб- так и прокси-сервер).

- **Тело сообщения** (англ. Message Body) — непосредственно данные сообщения. Обязательно должно отделяться от заголовков пустой строкой. Присутствие тела сообщения в запросе отмечается добавлением к заголовкам запроса поля заголовка Content-Length или Transfer-Encoding. Тело сообщения (message-body) может быть добавлено в запрос только когда метод запроса допускает тело объекта (entity-body).

Некоторые **основные методы HTTP**.

- GET ресурс HTTP/1.1: получение указанного ресурса.
- POST ресурс HTTP/1.1: получение указанного ресурса и передача вложенных входных данных на HTTP-сервер.
- HEAD ресурс HTTP/1.1: выполняется так же, как GET, но вместо ответа с полным содержимым URL-адреса сервер отправляет обратно только информацию заголовка, которая находится в разделе HTML..
- PUT ресурс HTTP/1.1: размещение ресурса на HTTP-сервере.
- DELETE ресурс HTTP/1.1: удаление ресурса с сервера.

Также HTTP поддерживает некоторые реже используемые **методы**:

- TRACE: используется для получения от сервера информации о «прыжках» (* ближайший маршрутизатор, маршрутизатор, находящийся на расстоянии одного прыжка), через которые прошел запрос.
- OPTIONS: для получения поддерживаемых сервером возможностей. На стороне клиента его можно использовать для изменения запроса в зависимости от возможностей, поддерживаемых сервером.

Некоторые основные коды состояния протокола HTTP

- **200 OK** (Успешно).
- **201 Created.** Это означает, что запрос был успешным и ресурс был создан. Код используется для подтверждения успеха запроса PUT или POST.
- **300 Moved Permanently.** Этот код ответа означает, что URL-адрес запрошенного ресурса был изменен навсегда.
- **400 Bad Request**. Запрос был сформирован неверно. Это происходит с запросами POST и PUT, когда данные не проходят проверку или имеют неправильный формат.
- **401 Unauthorized.** Эта ошибка указывает на то, что вам необходимо выполнить аутентификацию перед доступом к ресурсу.
- **404 Not Found.** Этот код показывает, что не удалось найти требуемый ресурс. 404 означает, что URL-адрес не распознается или запрашиваемого ресурса нет в указанном месте.

- **405 Forbidden.** Используемый метод HTTP не поддерживается для этого ресурса.
- **409 Conflict.** Код указывает на произошедший конфликт. Например, вы используете запрос PUT для создания одного и того же ресурса дважды.
- **500 Internal Server Error.** Как правило, ответ 500 используется, когда обработка запроса завершается неудачно из-за непредвиденных обстоятельств на стороне сервера.