**HTTP**

**HTTP** – протокол прикладного уровня передачи данных. Основа HTTP – технология «клиент-сервер», т.е. есть:

* **Потребители** (клиенты), которые инициируют соединение и посылают запрос;
* **Поставщики** (серверы), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом.

HTTP используется, чтобы получать информацию с сайтов, а также как «транспорт» для других протоколов прикладного уровня.

HTTP работает в основном с ресурсом, на который указывает URI в запросе клиента (хранящиеся на сервере файлы, логические объекты или что-то абстрактное). Особенность HTTP – возможность указать в запросе и ответе способ представления одного и того же ресурса по различным параметрам: формату, кодировке, языку и т. д. Благодаря возможности указания способа кодирования сообщения, клиент и сервер могут обмениваться двоичными данными, хотя данный протокол является текстовым.

HTTP — протокол прикладного уровня. Обмен сообщениями идёт по схеме «запрос-ответ». Для идентификации ресурсов HTTP использует глобальные URI. HTTP не сохраняет промежуточного состояния между парами «запрос-ответ». Компоненты, использующие HTTP, могут самостоятельно осуществлять сохранение информации о состоянии, связанной с последними запросами и ответами. Браузер, посылающий запросы, может отслеживать задержки ответов. Сервер может хранить IP-адреса и заголовки запросов последних клиентов. Но сам протокол не осведомлён о предыдущих запросах и ответах, и в нём не предусмотрена внутренняя поддержка состояния.

**Программное обеспечение**

ПО для работы с протоколом HTTP делятся на 3 категории:

* **Исходные серверы** – обрабатывают запросы. Поставщики услуг хранения и обработки информации. Основные реализации: [Apache](https://ru.wikipedia.org/wiki/Apache_HTTP_Server), [Internet Information Services](https://ru.wikipedia.org/wiki/Internet_Information_Services) (IIS), [nginx](https://ru.wikipedia.org/wiki/Nginx" \o "Nginx), [Google Web Server](https://ru.wikipedia.org/wiki/Google_Web_Server), [lighttpd](https://ru.wikipedia.org/wiki/Lighttpd" \o "Lighttpd).
* **Клиенты** – отправляют запросы. Конечные потребители услуг сервера. Часто ими являются браузеры, офлайн-браузеры, менеджеры закачек, программы для скачивания обновлений.
* **Прокси-серверы** – выполняют роль транспортных служб. Посредники. Основные реализации: [Squid](https://ru.wikipedia.org/wiki/Squid" \o "Squid), [UserGate](https://ru.wikipedia.org/wiki/UserGate" \o "UserGate), [Multiproxy](https://ru.wikipedia.org/w/index.php?title=Multiproxy&action=edit&redlink=1" \o "Multiproxy (страница отсутствует)), [Naviscope](https://ru.wikipedia.org/w/index.php?title=Naviscope&action=edit&redlink=1" \o "Naviscope (страница отсутствует)), [nginx](https://ru.wikipedia.org/wiki/Nginx" \o "Nginx).

**Структура протокола**

HTTP-сообщение состоит из 3-х частей, которые передаются в указанном порядке:

1. **Стартовая строка** – определяет тип сообщения. Присутствует обязательно.
2. **Заголовки** – характеризуют тело сообщения, параметры передачи и прочие сведения. Присутствует обязательно.
3. **Тело сообщения** – содержит данные сообщения. Обязательно должно отделяться от заголовков пустой строкой. Может отсутствовать.

**Стартовая строка**

Стартовые строки различаются для запроса и ответа:

* **Строка запроса:**

GET URI – для версии протокола 0.9;

Метод URI HTTP/Версия – для остальных версий.

* **Строка ответа сервера:**

HTTP/Версия КодСостояния Пояснение.

**Методы**

* **OPTIONS** – определяет возможности сервера или параметров соединения для конкретного ресурса.
* **GET** – выдает содержимое указанного ресурса.
* **HEAD** – аналогичен GET, но в ответе сервера отсутствует тело. Применяется для извлечения [метаданных](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%B0%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5), проверки наличия ресурса, и чтобы узнать, не изменился ли он с момента последнего обращения.
* **POST** – передает пользовательские данные заданному ресурсу.Многократное повторение одних и тех же запросов POST может возвращать разные результаты. В этом POST отличается от GET.
* **PUT** – загружает содержимое запроса на указанный в запросе URI.
* **PATCH** – аналогичен PUT, но применяется к фрагменту ресурса.
* **DELETE** –удаляет указанный ресурс.
* **TRACE** – возвращает полученный запрос так, что клиент может увидеть, какую информацию промежуточные серверы добавляют или изменяют в запросе.
* **CONNECT** – преобразует соединение запроса в прозрачный [TCP/IP](https://ru.wikipedia.org/wiki/TCP/IP)-туннель.

Если сервер не распознал метод, он возвращает статус 501. Если серверу метод известен, но он неприменим к конкретному ресурсу, возвращаем статус 405.

**Код состояния** – целое трехзначное число. 1-ая цифра указывает на класс состояния. 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**](https://ru.wikipedia.org/wiki/RFC)**.**

**Заголовки HTTP** – строки в HTTP-сообщении. Содержат пару параметр:значение. Формат заголовков соответствует общему формату заголовков текстовых сетевых сообщений ARPA (см. [**RFC 822**](https://tools.ietf.org/html/rfc822)). Заголовки нужно отделять от тела сообщения хотя бы одной пустой строкой.

**Тело сообщения –** если присутствует, используется для передачи тела объекта, связанного с запросом или ответом.

**Основные механизмы протокола**

* **Частичные GET** – запросы лишь указанного фрагмента. Их используют для [докачки файлов](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%BD%D0%B5%D0%B4%D0%B6%D0%B5%D1%80_%D0%B7%D0%B0%D0%BA%D0%B0%D1%87%D0%B5%D0%BA" \o "Менеджер закачек) и быстрого параллельного скачивания в нескольких потоках.Для получения фрагмента, клиент посылает серверу запрос с заголовком Range, указывая в нём необходимые [байтовые диапазоны](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B3%D0%BE%D0%BB%D0%BE%D0%B2%D0%BA%D0%B8_HTTP#%D0%91%D0%B0%D0%B9%D1%82%D0%BE%D0%B2%D1%8B%D0%B5_%D0%B4%D0%B8%D0%B0%D0%BF%D0%B0%D0%B7%D0%BE%D0%BD%D1%8B).
* **Условные GET** – GET, в котором сообщение запроса включает в себя поле заголовка If-Modified-Since. В ответ на «условный GET», тело запрашиваемого ресурса передаётся только если он изменялся после даты, указанной в заголовке If-Modified-Since.

**Согласование содержимого –** механизм автоматического определения необходимого ресурса при наличии нескольких разнотипных версий документа.

Есть 2 типа согласований:

* **Управляемое сервером** – если есть несколько версий ресурса, сервер может анализировать заголовки запроса клиента, чтобы выдать, по его мнению, наиболее подходящую.
* **Управляемое клиентом** – тип содержимого определяется только на стороне клиента.

Одновременно можно использовать оба типа.

**Множественное содержимое**

HTTP поддерживает передачу нескольких сущностей в пределах одного сообщения. Сущности можно передавать не только в виде одноуровневой последовательности, но в виде [иерархии](https://ru.wikipedia.org/wiki/%D0%98%D0%B5%D1%80%D0%B0%D1%80%D1%85%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0) с вложением элементов друг в друга.