Определение HTTP

**HTTP** (HyperText Transfer Protocol, дословно — «протокол передачи гипертекста») - расширяемый протокол прикладного уровня, используемый для доступа к ресурсам интернета.

**Гипертекст** - текст, в понятном для человека представлении, содержащий ссылки на другие ресурсы.

В обмене информацией по HTTP-протоколу принимают участие клиент и сервер по схеме:

- клиент запрашивает у сервера некоторый ресурс.

- сервер обрабатывает запрос и возвращает клиенту ресурс, который был запрошен.

Данные между клиентом и сервером в рамках работы протокола передаются с помощью HTTP-сообщений:

**Запросы (HTTP Requests)** — сообщения, которые отправляются клиентом на сервер, чтобы вызвать выполнение некоторых действий. Зачастую для получения доступа к определенному ресурсу. Основой запроса является HTTP-заголовок.

**Ответы (HTTP Responses)** — сообщения, которые сервер отправляет в ответ на клиентский запрос.

Структура HTTP запросов/ответов

Как запросы HTTP, так и ответы имеют следующую структуру:

- **стартовая строка (start line)** — используется для описания версии используемого протокола и другой информации — вроде запрашиваемого ресурса или кода ответа. Как можно понять из названия, ее содержимое занимает ровно одну строчку.

- **HTTP-заголовки (HTTP Headers)** — несколько строчек текста в определенном формате, которые либо уточняют запрос, либо описывают содержимое тела сообщения.

- **пустая строка**, которая сообщает, что все метаданные для конкретного запроса или ответа были отправлены.

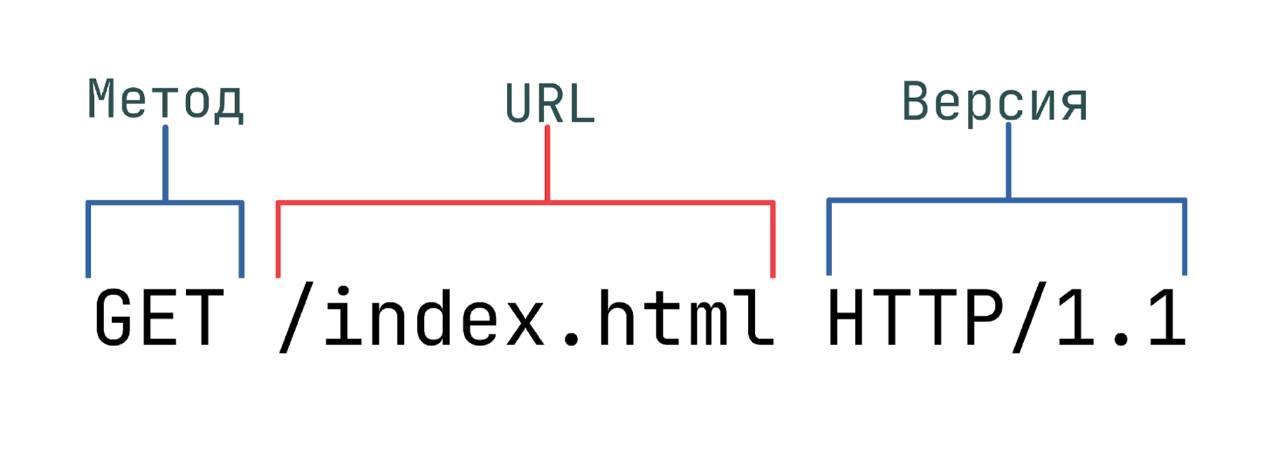
- **опциональное тело сообщения**, которое содержит данные, связанные с запросом, либо документ (например HTML-страницу), передаваемый в ответе.

Элементы стартовой строки

**Метод HTTP-запроса** - короткое слово, которое указывает, что конкретно нужно сделать с запрашиваемым ресурсом. Например, метод GET сообщает серверу, что пользователь хочет получить некоторые данные, а POST — что некоторые данные должны быть помещены на сервер.

**Цель запроса.** Представлена указателем ресурса URL, который состоит из протокола, доменного имени (или IP-адреса), пути к конкретному ресурсу на сервере. Дополнительно может содержать указание порта, несколько параметров HTTP-запроса и еще ряд опциональных элементов.

**Версия используемого протокола** (либо HTTP/1.1, либо HTTP/2), которая определяет структуру следующих за стартовой строкой данных.



Методы

Методы позволяют указать конкретное действие, которое мы хотим, чтобы сервер выполнил, получив наш запрос.

*Основные:*

**GET** – запрос данных/ресурса с сервера. Дополнительные данные могут быть переданы через строку запроса в составе URL (например ?param=value).

**POST** – отправка данных на сервер, среди которых текст, PDF-документы и другие типы данных в двоичном виде. В отличие от PUT, отправляющего весь ресурс в теле запроса, POST - запросы обычно отправляют только данные, которые обновляются.

**PUT** - используется для создания (размещения) новых ресурсов на сервере.

**DELETE** - позволяет удалить существующие ресурсы на сервере.

*Дополнительные:*

**HEAD** – метод позволит получить *только заголовки*, которые сервер вернул бы при GET-запросе к тому же ресурсу. Запрос с использованием данного метода обычно производится для того, чтобы узнать размер запрашиваемого ресурса перед его загрузкой.

**OPTIONS** - позволяет запросить информацию о сервере, в том числе информацию о допускаемых к использованию на сервере HTTP-методов.

**PATCH** - позволяет внести частичные изменения в указанный ресурс по указанному расположению.

URL

Получение доступа к ресурсам по HTTP-протоколу осуществляется с помощью указателя URL (Uniform Resource Locator).

URL представляет собой строку, которая позволяет указать запрашиваемый ресурс и еще ряд параметров.

Основные компоненты и строение URL:

**Поле Scheme** используется для указания используемого протокола, всегда сопровождается двоеточием и двумя косыми чертами (://).

**Host** - указывает местоположение ресурса, в нем может быть как доменное имя, так и IP-адрес.

**Port** – указывает номер порта, по которому следует обратиться к серверу. Оно начинается с двоеточия (:), за которым следует номер порта.

**Поле Path -** указывает на ресурс, к которому производится обращение.

**Поле Query String** начинается со знака вопроса (?), за которым следует пара «параметр-значение», между которыми расположен символ равно (=). В поле Query String могут быть переданы несколько параметров с помощью символа амперсанд (&) в качестве разделителя.

\*Обязательно следует указать только поля Scheme и Host.

http://example.com:8080/path/to/resource?param1=value1&param2=value2

* http://- схема (Scheme), указывающая на протокол.
* example.com: - хост (Host), указывающий на местоположение ресурса.
* 8080 - порт (Port), по которому следует обратиться к серверу.
* /path/to/resource - это путь (Path) к ресурсу.
* param1=value1&param2=value2 - строка запроса (Query String), где param1 и param2 -параметры, а value1 и value2 - их соответствующие значения.

Заголовки

**HTTP-заголовок** представляет собой строку формата «Имя-Заголовок:Значение», с двоеточием(:) в качестве разделителя. Название заголовка не учитывает регистр, однако, в названиях заголовков принято начинать каждое новое слово с заглавной буквы. Заголовок вместе со значениями занимает всегда одну строчку.

**Категории заголовков:**

* Общего назначения, которые применяются ко всему сообщению целиком.
* Заголовки запроса уточняют некоторую информацию о запросе, сообщая дополнительный контекст или ограничивая его некоторыми логическими условиями.
* Заголовки представления, которые описывают формат данных сообщения и используемую кодировку. Добавляются к запросу только в тех случаях, когда с ним передается некоторое тело.

Частые заголовки запроса

**Host** – указывает host, с которого запрашивается ресурс. В качестве значений могут использоваться как доменные имена, так и IP-адреса. На одном HTTP-сервере может быть размещено несколько различных веб-сайтов. Для обращения к какому-то конкретному требуется данный заголовок.

**User-Agent** - используется для описания клиента, который запрашивает ресурс. Содержит информацию о пользовательском окружении. Например, какой браузер используется в качестве клиента, его версию, а также операционную систему, на которой этот клиент работает.

**Refer** - используется для указания того, откуда поступил текущий запрос. Например, если вы решите перейти по какой-нибудь ссылке в этой статье, то вероятнее всего к запросу будет добавлен заголовок Refer: https://selectel.ru

**Accept** - позволяет указать, какой тип медиафайлов принимает клиент. В данном заголовке могут быть указаны несколько типов, перечисленные через запятую (‘ , ‘). А для указания того, что клиент принимает любые типы, используется следующая последовательность — \*/\*.

**Cookie**. Заголовок может содержать в себе одну или несколько пар «Куки-Значение» в формате cookie=value.

Куки представляют собой небольшие фрагменты данных, которые хранятся как на стороне клиента, так и на сервере, и выступают в качестве идентификатора.

Куки передаются вместе с запросом для поддержания доступа клиента к ресурсу. Помимо этого, куки могут использоваться и для других целей, таких как хранение пользовательских предпочтений на сайте и отслеживание клиентской сессии. Несколько кук в одном заголовке могут быть перечислены с помощью символа точка с запятой (‘ ; ‘), который используется как разделитель.

**Authorization**. Используется в качестве еще одного метода идентификации клиента на сервере. После успешной идентификации сервер возвращает токен, уникальный для каждого конкретного клиента. В отличие от куки, данный токен хранится исключительно на стороне клиента и отправляется клиентом только по запросу сервера.

Тело запроса

Завершающая часть HTTP-запроса — его тело. Не у каждого HTTP-метода предполагается наличие тела. Так, например, методам вроде GET, HEAD, DELETE, OPTIONS обычно не требуется тело. Некоторые виды запросов могут отправлять данные на сервер в теле запроса: самый распространенный из таких методов — POST.

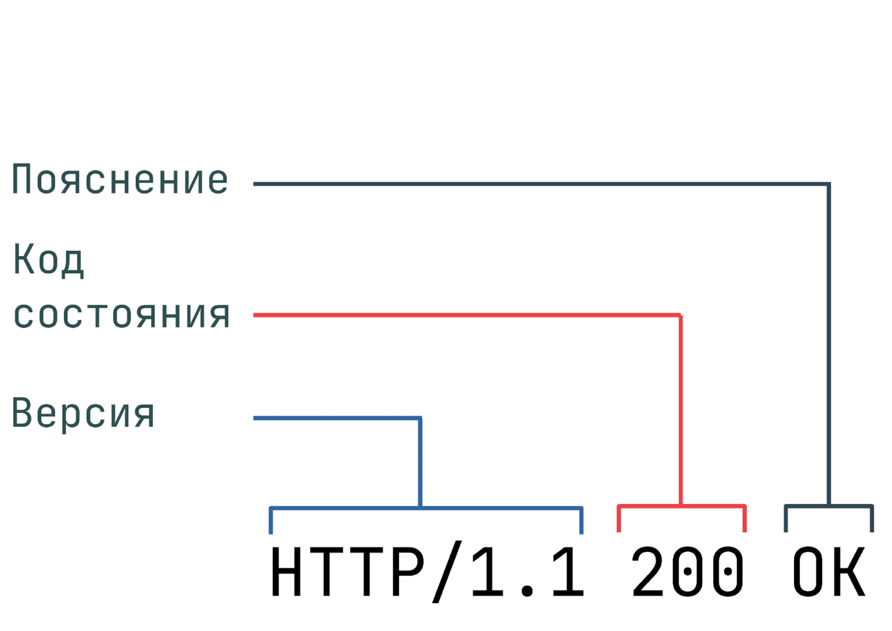
Ответы HTTP

HTTP-ответ является сообщением, которое сервер отправляет клиенту в ответ на его запрос. Его структура равна структуре HTTP-запроса: стартовая строка, заголовки и тело.

Строка статуса (Status line)

Стартовая строка HTTP-ответа называется строкой статуса (status line). На ней располагаются следующие элементы:

* версия протокола (HTTP/2 или HTTP/1.1).
* код состояния, который указывает, насколько успешно завершилась обработка запроса.
* пояснение — короткое текстовое описание к коду состояния. Используется исключительно для того, чтобы упростить понимание и восприятие человека при просмотре ответа.



Коды состояния и текст статуса

**1xx** - коды из данной категории носят исключительно информативный характер и никак не влияют на обработку запроса.

**2xx** - коды состояния из этой категории возвращаются в случае успешной обработки клиентского запроса.

**3xx -** коды, которые возвращаются, если серверу нужно перенаправить клиента.

**4xx** - коды данной категории означают, что на стороне клиента был отправлен некорректный запрос. Например, клиент в запросе указал не поддерживаемый метод или обратился к ресурсу, к которому у него нет доступа.

**5xx** - ответ с кодами из этой категории приходит, если на стороне сервера возникла ошибка.

Заголовки ответа

**Response Headers,** или заголовки ответа, используются для того, чтобы уточнить ответ, и никак не влияют на содержимое тела. Они существуют в том же формате, что и остальные заголовки, а именно «Имя-Значение» с двоеточием (:) в качестве разделителя.

**Server.**

Пример Server: ngnix. Содержит информацию о сервере, который обработал запрос.

**Set-Cookie**

Пример: Set-Cookie:PHPSSID=bf42938f. Содержит куки, требуемые для идентификации клиента. Браузер парсит куки и сохраняет их в своем хранилище для дальнейших запросов.

**WWW-Authenticate**

Пример: WWW-Authenticate: BASIC realm=»localhost». Уведомляет клиента о типе аутентификации, который необходим для доступа к запрашиваемому ресурсу.

Тело ответа

Последней частью ответа является его тело. Оно не является обязательным. Например, у кодов «201 Created» или «204 No Content» тело отсутствует, так как достаточную информацию для ответа на запрос они передают в заголовке.

HTTPs

HTTPs (HyperText Transfer Protocol, secure) является расширением HTTP-протокола, который позволяет шифровать отправляемые данные, перед тем как они попадут на транспортный уровень. Данный протокол по умолчанию использует порт 443.