**ТЕСТОВОЕ ЗАДАНИЕ. ЯНДЕКС. “HTTP-ПРОТОКОЛ”**

**Что такое HTTP-протокол?**

**HTTP** (Hyper Text Transfer Protocol) — базовый сетевой протокол, который обеспечивает доступ к различным ресурсам в Интернете, например html-документам.

Клиенты и серверы взаимодействуют между собой обмениваясь индивидуальными сообщениями. Есть два вида таких сообщений:

* **HTTP-запрос (request)**
* **HTTP-ответ (response)**

**Структура HTTP-сообщения**

1. Стартовая строка (определяет тип сообщения)
2. HTTP заголовки
3. Тело сообщения (опционально)

Структура стартовой строки HTTP-запроса и ответа отличаются друг от друга, HTTP заголовки обязательно присутствуют в обоих сообщениях, а тело сообщения может отсутствовать. Далее рассмотрим HTTP-сообщения более подробно.

**Что такое HTTP-запрос?**

HTTP-запрос ­– сообщение, которое отправляется клиентом на сервер, чтобы вызвать его реакцию в виде ответа. Каждый запрос, выполненный через Интернет, несет в себе серию закодированных данных, которые содержат различные типы информации.

Стартовая строка HTTP-запроса содержит 3 элемента:

* HTTP-метод.
* URL.
* Версия HTTP.

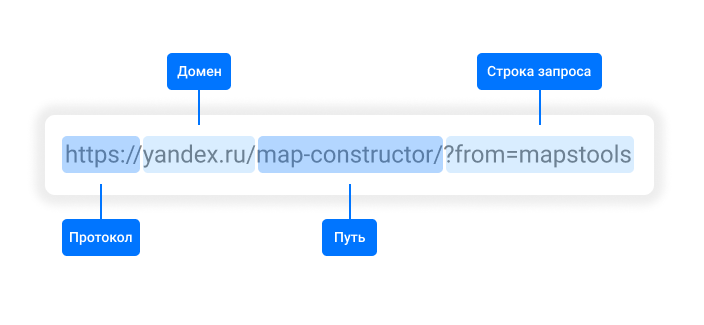
**Что такое HTTP-метод?**

HTTP-метод, также называемый HTTP глаголом, указывает на действие, которое HTTP-запрос ожидает от запрашиваемого сервера. Наиболее распространенные виды HTTP-методов:

* **GET.** Запрашивает представление указанного ресурса. Запросы с использованием этого метода могут только извлекать данные.
* **HEAD**. Запрашивает ресурс так же, как и метод GET, но без тела ответа.
* **POST.** Используется для отправки сущностей к определённому ресурсу. Часто вызывает изменение состояния или какие-то побочные эффекты на сервере.
* **PUT.** Заменяет все текущие представления ресурса данными запроса.
* **DELETE.** Удаляет указанный ресурс**.**

**Что такое URL?**

URL (Uniform Resource Locator) – абсолютный путь, порт и домен, на котором расположен ресурс.



**Для чего указывается версия HTTP?**

Версия HTTP определяет то, какую структуру имеет оставшаяся часть сообщения (заголовки и тело сообщения), а также указывает какую версию предполагается использовать для ответа. Версии HTTP:

* **HTTP/0.9**. Простейшая реализация протокола, позволяющая только получать WEB – страницы
* **HTTP/1.0**. Версия включает в себя HTTP заголовки.
* **HTTP/1.1**. Улучшенная версия протокола 1.0, поддерживающая методы постоянных TCP-сессий, а также позволяющая отправку нескольких HTTP-запросов одновременно
* **HTTP/2.** В отличие от предыдущих версий, протокол является бинарным, эффективнее использует сетевые ресурсы и имеет более высокие показатели производительности.

Современные браузеры обеспечивают полную совместимость данных версий, поэтому в случае отправки запроса версии 1.0 и получения ответа 1.1, данные будут успешно обработаны.

**Что такое заголовки HTTP-запроса?**

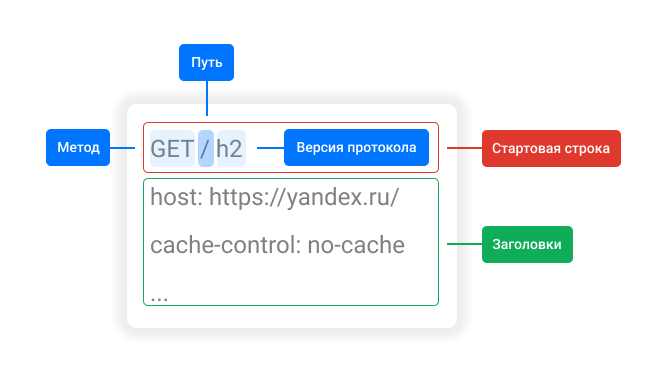
Заголовки HTTP-запроса используются для точного описания загружаемого ресурса или поведения сервера или клиента. Заголовки являются частью каждого HTTP-запроса и содержат текстовую информацию, которая хранится в парах вида: ключ-значение. Заголовки HTTP-запроса подразделяются на:

* **Основные заголовки (General headers).** Относятся к сообщению в целом
* **Заголовки запроса (Request headers).** Уточняют запрос или накладывают ограничения на условия.
* **Заголовки сущности (Entity headers).** Относятся к телу сообщения, такие заголовки отсутствуют, если нет тела.

**Тело HTTP-запроса**

Является последним и необязательным элементом HTTP-запроса. Так, для методов вида GET, HEAD, DELETE, или OPTIONS тело запроса не требуется, однако используется, в случае, когда данные отправляется на сервер для обновления методом POST.

**Пример HTTP-запроса:**



**Что такое HTTP-ответ?**

HTTP-ответ – информация, которую получает клиент в ответ на HTTP-запрос. Стартовая строка HTTP-ответа также содержит 3 элемента:

* Версия HTTP
* Код состояния HTTP
* Пояснение состояния (короткое пояснение кода состояния, например, для ошибки 404: “Not Found”)

**Что такое** **код состояния HTTP?**

Код состояния HTTP представляет собой трехзначный код, который чаще всего используется для того, чтобы определить успешно ли выполнен запрос или нет. Набор кодов состояния является стандартом, кроме того, их можно разделить на 5 логических групп:

1. **100–199**. Информационные коды — запрос получен, продолжается обработка.
2. **200–299**. Успешные коды — запрос был успешно получен, понят и выполнен.
3. **300–399**. Коды перенаправления — для выполнения запроса должны быть предприняты дальнейшие действия агента пользователя (программы, принимающей ответ на запрос).
4. **400–499.** Коды ошибок клиента — запрос имеет некорректный синтаксис или не может быть выполнен. Также указывает, является ли ситуация временной или постоянной.
5. **500–599**. Коды ошибок сервера — сервер не в состоянии выполнить допустимый запрос или выполнил его с ошибкой.

**Состав** **заголовков HTTP-ответа**

Заголовки HTTP-ответа похожи по своей структуре на заголовки для запросов, имеют такой же формат записи (пара ключ-значение) и также состоят из одной строки. Поскольку существует большое количество заголовков HTTP-ответа их разделяют на 3 группы:

* **Основные заголовки (General headers).** Относятся к сообщению в целом.
* **Заголовки ответа (Response headers).** Содержат дополнительные сведения о сервере.
* **Заголовки сущности (Entity headers).** Относятся к телу сообщения, такие заголовки отсутствуют, если нет тела.

**Тело HTTP-ответа**

Также, как и в запросе, тело является последним и необязательным элементом. Включение тела зависит не только от используемого метода, но и от кода состояний. Например, ответы, которые содержат коды состояния 1ХХ, 204, 304 не должны содержать тела сообщения, все другие ответы обычно содержат тело сообщения.

**HTTPS и его отличия от HTTP**

HTTPS (HTTP Secure) – это защищенная версия протокола HTTP, которая использует технологии SSL или TLS для шифрования всех соединений между клиентом и сервером. Такое защищённое соединение позволяет клиентам безопасно обмениваться конфиденциальными данными с сервером, например, для банковских операций или онлайн-покупок.

**Назначение [35]**

Более подробно взаимодействие клиент-сервер строится по следующей схеме:

Когда клиент хочет взаимодействовать с сервером, являясь конечным сервером или промежуточным прокси, он выполняет следующие шаги:

1. Открытие TCP соединения: TCP-соедиенение будет использоваться для отправки запроса или запросов, и получения ответа. Клиент может открыть новое соединение, переиспользовать существующее, или открыть несколько TCP-соединений к серверу.
2. Отправка HTTP-сообщения: HTTP-собщения (до HTTP/2) -- человеко-читаемо. Начиная с HTTP/2, простые сообщения инкапсилуруются во фреймы, делая невозможным их чтения напрямую, но принципиально остаются такими же.

*GET / HTTP/1.1*

*Host: developer.mozilla.org*

*Accept-Language: fr*

1. Читает ответ от сервера:

*HTTP/1.1 200 OK*

*Date: Sat, 09 Oct 2010 14:28:02 GMT*

*Server: Apache*

*Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT*

*ETag: "51142bc1-7449-479b075b2891b"*

*Accept-Ranges: bytes*

*Content-Length: 29769*

*Content-Type: text/html*

*<!DOCTYPE html... (here comes the 29769 bytes of the requested web page)*

1. Закрывает или переиспользует соединение для дальнейщих запросов.

То же самое, но другими словами

HTTP следует заданному циклу всякий раз, когда посылает запрос:[[2]](https://www.speedcheck.org/ru/wiki/http/#fn2)

1. Браузер запросит HTML-страницу. Затем сервер возвращает HTML-файл с хоста.1
2. Браузер запросит таблицу стилей. Затем сервер возвращает файл CSS.
3. Браузер запрашивает изображение в формате JPG. Сервер возвращает файл JPG.
4. Браузер запросит код JavaScript (язык программирования). После этого сервер возвращает JS-файл.
5. Браузер запрашивает различные формы данных. Сервер возвращает данные в виде XML или JSON файлов.

**Устройство []**

**Структура HTTP-сообщения**

1. **Стартовая строка** ([англ.](https://ru.m.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Starting line*) — определяет тип сообщения;
   1. Стартовая строка запроса:
      1. HTTP-метод, обычно глагол подобно GET, POST или существительное, как OPTIONS или HEAD, определяющее операцию, которую клиент хочет выполнить. Обычно, клиент хочет получить ресурс (используя GET) или передать значения HTML-формы (используя POST), хотя другие операция могут быть необходимы в других случаях.[URI](https://ru.m.wikipedia.org/wiki/URI) определяет путь к запрашиваемому документу.
      2. Путь к ресурсу: URL ресурсы лишены элементов, которые очевидны из контекста, например без protocol (http://), domain (здесь developer.mozilla.org), или TCP port (здесь 80).
      3. Версия ([англ.](https://ru.m.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Version*) — пара разделённых точкой [цифр](https://ru.m.wikipedia.org/wiki/%D0%90%D1%80%D0%B0%D0%B1%D1%81%D0%BA%D0%B8%D0%B5_%D1%86%D0%B8%D1%84%D1%80%D1%8B). Например: 1.0.
   2. Стартовая строка ответа
      1. Версию HTTP-протокола.
      2. [HTTP код состояния](https://developer.mozilla.org/en-US/docs/Web/HTTP/Status), сообщающий об успешности запроса или причине неудачи.
      3. Сообщение состояния -- краткое описание кода состояния.
      4. HTTP [заголовки](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers), подобно заголовкам в запросах.
      5. Опционально: тело, содержащее пересылаемый ресурс.
2. **Заголовки** ([англ.](https://ru.m.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Headers*) — характеризуют тело сообщения, параметры передачи и прочие сведения;
3. **Тело сообщения** ([англ.](https://ru.m.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Message Body*) — непосредственно данные сообщения. Обязательно должно отделяться от заголовков пустой строкой.

**Особенности []**

1. **HTTP – прост**

Даже с большей сложностью, введенной в HTTP/2 путем инкапсуляции HTTP-сообщений в фреймы, HTTP, как правило, прост и удобен для восприятия человеком. HTTP-сообщения могут читаться и пониматься людьми, обеспечивая более легкое тестирование разработчиков и уменьшенную сложность для новых пользователей.

1. **HTTP - расширяемый**

Введенные в HTTP/1.0 [HTTP-заголовки](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers) сделали этот протокол легким для расширения и экспериментирования. Новая функциональность может быть даже введена простым соглашением между клиентом и сервером о семантике нового заголовка.

1. **HTTP не имеет состояния, но имеет сессию**

Не существует связи между двумя запросами, которые последовательно выполняются по одному соединению. Из этого немедленно следует возможность проблем для пользователя, пытающегося взаимодействовать с определенной страницей последовательно, например, при использовании корзины в электронном магазине. Но хотя ядро HTTP не имеет состояния, куки позволяют использовать сессии с сохранением состояния. Используя расширяемость заголовков, куки добавляются к рабочему потоку, позволяя сессии на каждом HTTP-запросе делиться некоторым контекстом, или состоянием.

1. **HTTP же устанавливает отдельную TCP-сессию на каждый запрос;**

Такое поведение отлично от большинства протоколов. В ходе сессии один раз происходит авторизация, и дальнейшие действия выполняются в контексте этой авторизации в более поздних версиях HTTP было разрешено делать несколько запросов в ходе одной TCP-сессии, но браузеры обычно запрашивают только страницу и включённые в неё объекты (картинки, каскадные стили и т. п.), а затем сразу разрывают TCP-сессию. Для поддержки авторизованного (неанонимного) доступа в HTTP используются [cookies](https://ru.wikipedia.org/wiki/Cookies" \o "Cookies); причём такой способ авторизации позволяет сохранить сессию даже после перезагрузки клиента и сервера.

При доступе к данным по FTP или по файловым протоколам тип файла (точнее, тип содержащихся в нём данных) определяется по расширению имени файла, что не всегда удобно. HTTP перед тем, как передать сами данные, передаёт заголовок «Content-Type: тип/подтип», позволяющий клиенту однозначно определить, каким образом обрабатывать присланные данные. Это особенно важно при работе с [CGI](https://ru.wikipedia.org/wiki/CGI)-скриптами, когда расширение имени файла указывает не на тип присылаемых клиенту данных, а на необходимость запуска данного файла на сервере и отправки клиенту результатов работы программы, записанной в этом файле (при этом один и тот же файл в зависимости от аргументов запроса и своих собственных соображений может порождать ответы разных типов — в простейшем случае картинки в разных форматах).

1. Кроме того, HTTP позволяет клиенту прислать на сервер параметры, которые будут переданы запускаемому CGI-скрипту. Для этого же в [HTML](https://ru.wikipedia.org/wiki/HTML) были введены формы.
2. протокол HTTP не предполагает использование шифрования для передачи информации. Тем не менее, для HTTP есть распространённое расширение, которое реализует упаковку передаваемых данных в криптографический протокол **SSL** или **TLS**.

Название этого расширения — **HTTPS** (*HyperText Transfer Protocol Secure*). Для HTTPS-соединений обычно используется TCP-порт 443. HTTPS широко используется для защиты информации от перехвата, а также, как правило, обеспечивает защиту от атак вида **man-in-the-middle** — в том случае, если сертификат проверяется на клиенте, и при этом приватный ключ сертификата не был скомпрометирован, пользователь не подтверждал использование неподписанного сертификата, и на компьютере пользователя не были внедрены сертификаты центра сертификации злоумышленника.