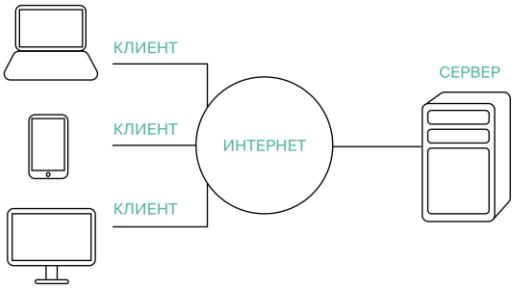
Клиент-серверная архитектура

Веб-приложение устроено по определённым принципам. Они определяют, из каких элементов состоит приложение и как они связаны.
Такие принципы называют **архитектурой**. Это способ организации работы приложения.
Самая распространённая архитектура веб-приложений — **клиент-серверная**.



Клиент, сервер и интернет — элементы клиент-серверной архитектуры. Клиент и сервер работают по отдельности: клиент отвечает за взаимодействие с пользователем, сервер — за логические операции, вычисления и хранение данных, а интернет, или сеть — за связь клиента и сервера.
Клиент — это система, которая связывается с сервером и запрашивает нужную пользователю информацию.
Клиент преобразует твои действия в запросы и отправляет на сервер.
Сервер — система, которая обрабатывает запросы клиента и формирует ответ. Например, присылает информацию о видео: лайки, комментарии, количество просмотров.
Интернет, или **Сеть** — система связанных между собой устройств, которая помогает клиенту и серверу обмениваться данными.
Фронтенд (frontend) — видимая часть приложения. С ней пользователь взаимодействует: например, нажимает на кнопки или вводит текст. В модели «клиент-сервер» фронтенд — это код, который обрабатывается на стороне клиента.
Бэкенд (backend) — скрытая часть приложения. Она отвечает за вычисления, логику, хранение данных. В модели «клиент-сервер» бэкенд — это код, который работает на удалённом сервере.
URL (Uniform Resource Locator — унифицированный указатель ресурса) — это адрес веб-ресурса. URL показывает, где находится веб-приложение, веб-страница или фрагмент веб-страницы и как к ним обратиться.

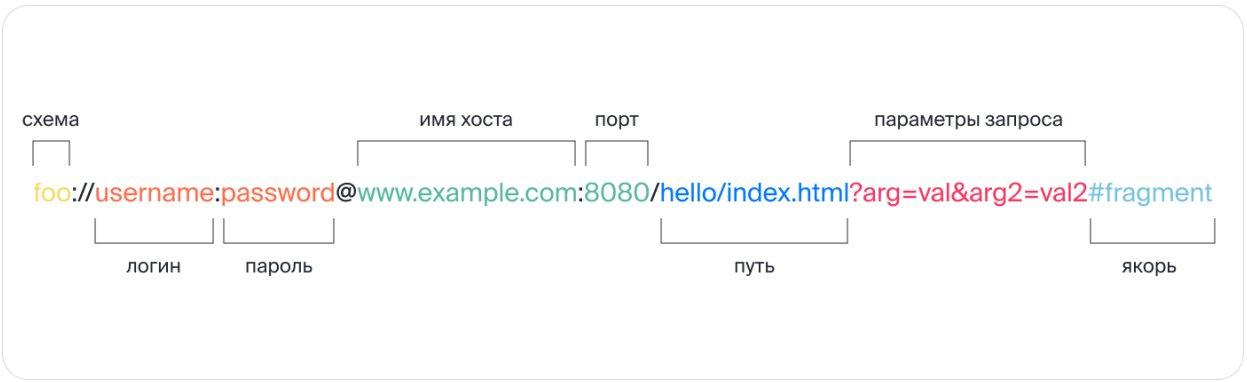


Схема и имя хоста — обязательные элементы URL.

Схема (scheme) — протокол, по которому передаются данные. Например: HTTP и HTTPS.
Имя хоста**:**порт (hostname:port) — доменное имя или IP-адрес сервера, к которому обращается пользователь.

IP-адрес — это уникальный идентификатор сервера, на котором находится нужная информация.
Протокол передачи данных — набор правил, по которым устройства обмениваются информацией.
HTTP (HyperText Transfer Protocol — протокол передачи гипертекста). Сейчас по HTTP передаётся не только гипертекст, но и другие данные: картинки, аудио, видео.
HTTPS (HyperText Transfer Protocol Secure — протокол защищённой передачи гипертекста). Он шифрует соединение по криптографическим протоколам: так клиент и сервер смогут безопасно передавать сообщения друг другу.

По протоколу HTTP фронтенд передаёт запрос (request), а бэкенд — ответ (response).
Запрос структурирован по правилам передачи данных HTTP. Он состоит из трёх блоков:

- стартовая строка,
- заголовки,
- тело сообщения.

Стартовая строка (start line) состоит из трёх элементов: метода, пути до ресурса и версии протокола.
Например: POST / search / HTTP/1.1 или GET / HTTP/1.1.

- **Метод** (method) указывает действие: бэкенд принимает его в обработку.
  Самые распространённые:
  GET – запросить у бэкенда данные по определенному адресу
  POST – Отправить данные на бэкенд
  PUT – Изменить данные на бэкенде
  DELETE – Удалить данные на бэкенде

- **Путь до ресурса** — адрес, по которому фронтенд отправляет запрос на бэкенд.
- **Версия протокола** — номер версии HTTP. Сейчас применяют версию HTTP/1.1.

Тело сообщения — это данные, которые передаёт фронтенд. (форматы JSON, XML, текст…)

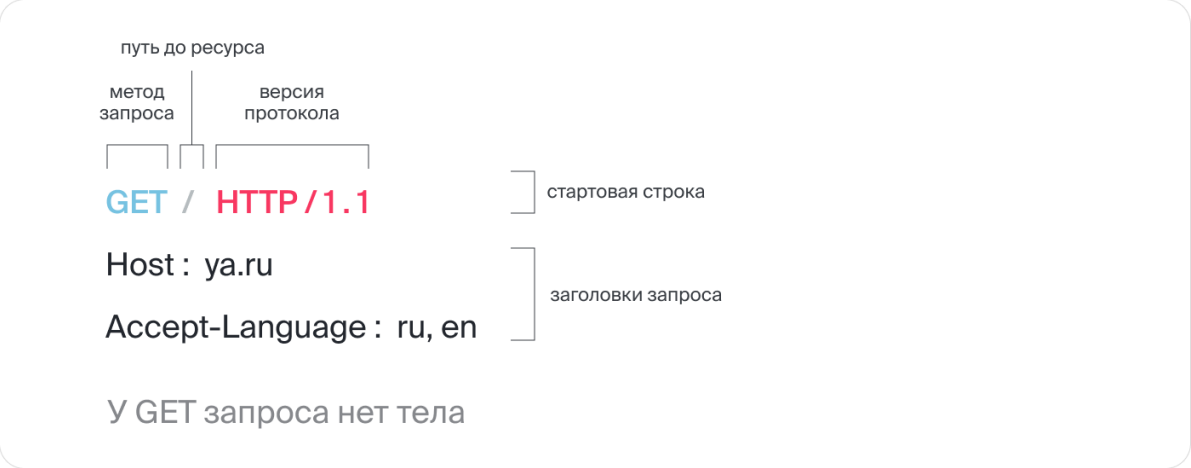
Пример запроса:

```
POST /subscribe/ HTTP/1.1 // POST — запись данных методом POST; /subscribe — запрос обращается к ресурсу, который отвечает за подписки; HTTP/1.1 — протокол HTTP версии 1.1;

Host: kinoart.ru // kinoart.ru — хост, к которому фронтенд обращается с запросом;
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.106 YaBrowser/21.6.0.620 Yowser/2.5 Safari/537.36 // здесь информация о браузерах, операционной системе и версиях ПО клиента;
Accept-Language: ru,en // ru,en — допустимые языки в ответе;
Connection: keep-alive // keep-alive — постоянное соединение с сервером;

Content-type: application/json // данные в формате JSON

Request body: // тело сообщения
{
"surname": "Иванов",
"name": "Иван",
"age": 28,
"address": "г. Москва, ул. Большая Садовая, дом 10, кв. 50"
}
```



HTTP-ответ включает:

- строку состояния,
- заголовки,
- тело сообщения.

1xx – Информационные сообщения
2xx – Сообщение об успехе
3xx – Перенаправление
4xx – Клиентские ошибки
5xx – Ошибки сервера

Например:

- 102 — запрос принят, но обработка ещё не завершена
- 200 — запрос обработан успешно
- 302 — запрошенный ресурс временно доступен по другому адресу
- 403 — у клиента недостаточно прав, чтобы получить доступ к данному ресурсу.
- 404 — страница, запрошенная клиентом, не найдена на сервере
- 500 — внутренняя ошибка сервера.
- 503 — сервис временно недоступен; сервер не может выполнять запросы.
- 504 — время ожидания ответа от сервера истекло.

Остальные можно уточнить тут: https://datatracker.ietf.org/doc/html/rfc2068#section-6.1.1

Пример запроса – ответа:

GET / HTTP/1.1 // GET — запрос данных методом GET; / — запрос на главной странице; HTTP/1.1 — протокол HTTP версии 1.1;

Host: ya.ru{target="_blank"} // ya.ru — хост, к которому фронтенд обращается с запросом;
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.106 YaBrowser/21.6.0.620 Yowser/2.5 Safari/537.36 // здесь информация о браузерах, операционной системе и версиях ПО клиента;
Accept-Language: ru,en // ru,en — допустимые языки в ответе;
Connection: keep-alive // keep-alive — постоянное соединение с сервером;

HTTP/1.1 200 Ok // HTTP/1.1 — ответ по протоколу HTTP версии 1.1; 200 Ок — запрос обработан успешно;

Date: Tue, 13 Jul 2021 08:10:41 GMT // Tue, 13 Jul 2021 08:10:41 GMT — дата создания ответа;
Content-Type: text/html; charset=UTF-8 // text/html — тип данных в ответе; charset=UTF-8 — UTF-8, единый стандарт текста в интернете;

Response Body: // в этом примере тело ответа содержит веб-страницу;
<!DOCTYPE html><html class="i-ua_js_no i-ua_css_standart i-ua_browser_yandexbrowser i-ua_browser-engine_webkit m-stat i-ua_browser_desktop i-ua_platform_windows" lang="ru...

Пример ещё когда Код и текст состояния — 200 Ок, тип данных — application/json:

HTTP/1.1 200 Ok

Connection: keep-alive
Content-Length: 154
Content-Type: application/json
Date: Wed, 04 Aug 2021 07:08:27 GMT
Server: nginx/1.10.3

Response body:
{"orderInfo":{"tariff":"Рабочий","features":{},"phone":"+71231231212","payment":"cash","comment":"Захватите виски"},"eta":23,"id":77}