

HTTP

HTTP база

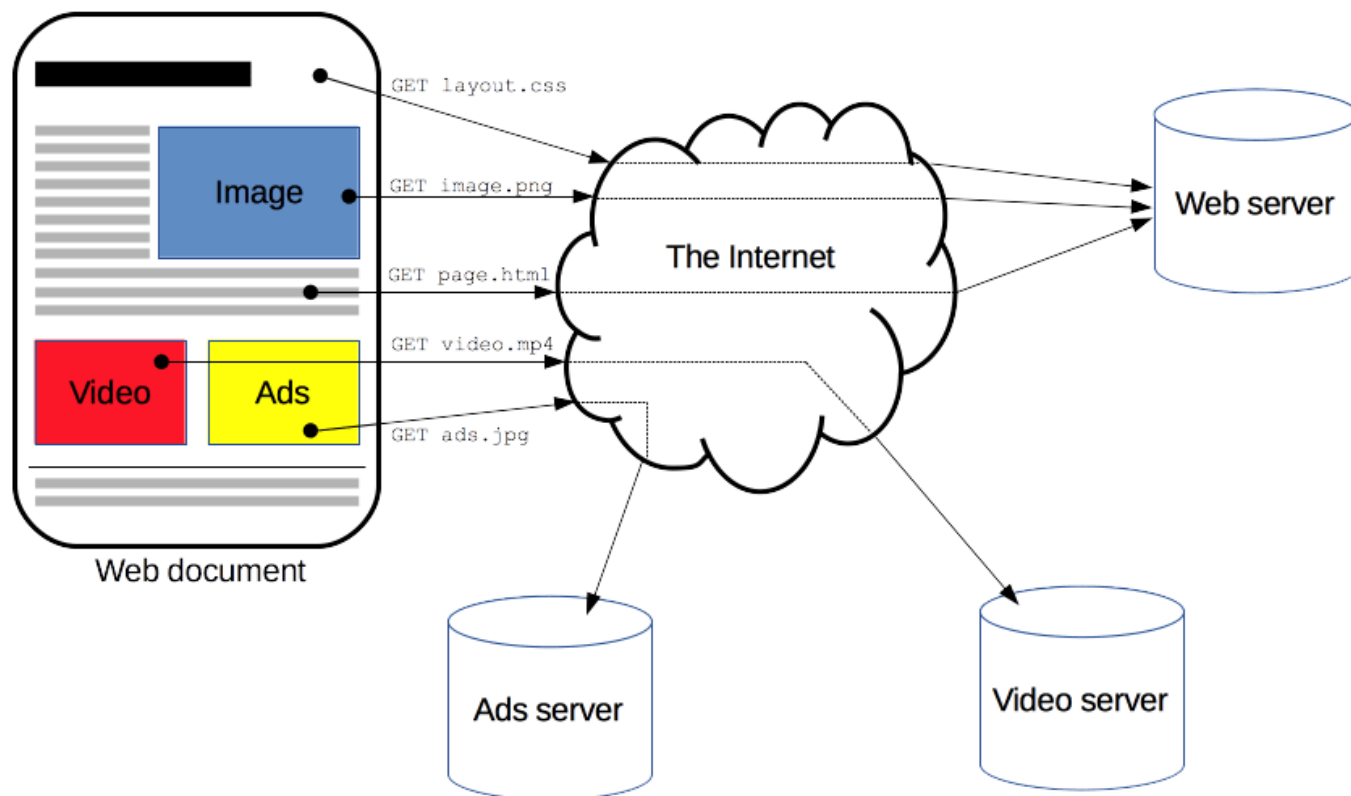
HTTP (Hypertext Transfer Protocol) - это протокол передачи гипертекста, который играет ключевую роль в коммуникации между клиентскими приложениями и веб-серверами. Является протоколом прикладного уровня, который определяет стандарты и правила для передачи данных в виде гипертекстовых документов. Гипертекстовые документы могут содержать ссылки на другие документы, образуя таким образом веб-сайты и страницы, которые мы привыкли видеть в браузерах.

Протокол основан на клиент-серверной архитектуре, где клиент отправляет запросы серверу, а сервер отвечает на эти запросы, предоставляя требуемую информацию или выполняя определенные действия. Запросы и ответы состоят из заголовков и, возможно, тела, которые содержат сами данные.

Принцип работы HTTP

Процесс обмена данными между клиентом и сервером в HTTP выглядит следующим образом:

1. Клиент инициирует соединение с сервером, отправляя HTTP-запрос.
2. Сервер принимает запрос и анализирует его.
3. Сервер генерирует HTTP-ответ, который содержит код состояния, заголовки и, возможно, тело с данными.
4. Ответ отправляется клиенту.
5. Клиент получает ответ и обрабатывает его в соответствии с кодом состояния и данными.



Клиенты и серверы взаимодействуют, обмениваясь одиночными сообщениями (а не потоком данных). Сообщения, отправленные клиентом, обычно веб-браузером, называются *запросами*, а сообщения, отправленные сервером, называются *ответами*.

Коды состояния

Коды состояния представляют собой трехзначные числа, которые указывают на результат обработки запроса сервером. Некоторые распространенные коды состояния включают:

- 200 OK: Запрос успешно выполнен, и сервер возвращает запрашиваемые данные.
- 404 Not Found: Запрашиваемый ресурс не найден на сервере.
- 500 Internal Server Error: Внутренняя ошибка сервера, которая мешает выполнить запрос.

Типы запросов

HTTP определяет несколько типов запросов, которые клиент может отправить серверу для выполнения определенных действий или получения данных. Некоторые из основных типов запросов включают:

- GET: Используется для получения данных с сервера. Клиент отправляет запрос на определенный ресурс и получает данные, связанные с этим ресурсом в ответе сервера.
- POST: Используется для отправки данных на сервер для обработки или создания нового ресурса. Клиент отправляет данные в теле запроса, и сервер обрабатывает эти данные соответствующим образом.

- PUT: Используется для обновления данных на сервере. Клиент отправляет данные, указывая идентификатор ресурса, который требуется обновить, и сервер обновляет соответствующий ресурс.
- DELETE: Используется для удаления ресурса на сервере. Клиент отправляет запрос на удаление указанного ресурса, и сервер удаляет его, если это возможно.
- PATCH: Используется для частичного обновления ресурса на сервере. Клиент отправляет только измененные или обновленные данные, и сервер применяет эти изменения к ресурсу.
- OPTIONS: Используется для получения информации о возможностях сервера и поддерживаемых методах запросов.

Каждый тип запроса имеет свою уникальную функцию и используется в различных сценариях. Они позволяют клиентам взаимодействовать с серверами и выполнять различные операции, такие как получение данных, отправка данных, обновление ресурсов и т. д.

Более показательные примеры:

GET запрос:

```
GET /search?query=example&page=1 HTTP/1.1
Host: example.com
```

В примере, мы отправляем GET запрос на `example.com/search` с параметрами `query=example` и `page=1`. Весь запрос, включая параметры, передается через URL.

Пример ответа сервера на GET запрос (HTTP 1.1 200 OK):

```
HTTP/1.1 200 OK
Date: Mon, 20 May 2023 10:30:00 GMT
Server: Apache/2.4.7 (Ubuntu)
Content-Type: text/html; charset=utf-8
Content-Length: 342
```

```
<!DOCTYPE html>
<html>
<head>
<title>Пример страницы</title>
</head>
<body>
<h1>Привет, мир! </h1>
</body>
</html>
```

POST запрос:

```
POST /submit HTTP/1.1
Host: example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 25

username=example&password=123
```

Примере, мы отправляем POST запрос на `example.com/submit`. Данные запроса, такие как `username` и `password`, передаются в теле запроса. Заголовки `Content-Type` и `Content-Length` указывают на тип данных и длину тела запроса.

Пример ответа сервера на POST запрос (HTTP 1.1 201 Created):

```
HTTP/1.1 201 Created
Date: Mon, 20 May 2023 10:45:00 GMT
Server: Apache/2.4.7 (Ubuntu)
Content-Type: application/json
Content-Length: 54

{"message": "Данные успешно созданы", "id": 12345}
```

В обоих примерах, первая строка представляет собой статус-линию ответа, которая содержит версию протокола (HTTP/1.1), код состояния (200 или 201) и текстовое описание состояния (OK или Created). Затем следуют заголовки, которые предоставляют дополнительную информацию о содержимом и свойствах ответа. Каждый заголовок представлен в формате "Имя: Значение".

После заголовков идет пустая строка, а затем следует тело ответа, которое содержит фактическое содержимое ответа сервера. Тело ответа может быть представлено в различных форматах, таких как HTML, JSON, XML и других, в зависимости от типа контента, указанного в заголовке "Content-Type".

Коды состояния HTTP указывают на успешность или ошибку выполнения запроса, а заголовки и тело ответа предоставляют дополнительную информацию и данные, возвращаемые сервером в ответ на запрос клиента.

Основное отличие между GET и POST методами заключается в том, что GET используется для получения данных с сервера через URL, в то время как POST используется для отправки данных в теле запроса. GET запросы могут быть закладками или расшаренными ссылками, так как параметры видны в URL, в то время как POST запросы обычно используются для отправки конфиденциальной или чувствительной информации, такой как пароли или данные кредитных карт.

Revision #4

Created 10 May 2023 11:31:34 by Maru

Updated 10 May 2023 11:58:21 by Maru