**URI-адреса**

Универсальный идентификатор ресурса.

URI позволяет идентифицировать любой информационный объект в сети

Текущая структура и синтаксис URI регулируется стандартом RFC 3986

URI-адрес представляет собой последовательность символов. Начинается со схемы. Схема может дополнительно ограничивать синтаксис и семантику используемых в этой схеме идентификаторов.

Основные принципы URI

- Полнота

- Расширяемость

- Читаемость

- Понятие ресурса используется в самом общем смысле

Синтаксис URI

URI состоит из последовательности символов в нижнем регистре.

URI организован иерархически из компонент, перечисленных в порядке убывания значимости слева направо

Базовый синтаксис URI включает 5 компонент: схема (schema), основание(authority), путь (path), запрос (query), фрагмент (fragment)

<http://example.com:8042/over/tree?name=ferret#nose>

Основные схемы доступа

ftp, http, https, mailto, news, nntp, telnet, file

Компонента “Основание”

Компонента основание следует за символами //, завершается символом /, знаком вопроса ?, символом #, или просто окончанием URI. Не является обязательной

В состав компоненты могут входить имя пользователя (логин), адрес хоста, номер порта:

Authority = [userinfo:password “@”] Host [“:” port]

Субкомпонента host: ipv6-адрес в квадратных скобка, ipv4-адрес или DNS-имя.

Компоненты URI

Путь и имя файла, определяют каталог страниц у запрошенного ресурса. Необязательно адресует физический файл на сервере, может создаваться динамически.

Например: /training/generic.asp

Запрос. Начинается символом ?

Синтаксис переменная = значение. Несколько компонентов запроса могут объединяться символом &.

Например: ?pageid=1078&country=DACH

Фрагмент. Начинается с #. Используется для ссылок внутри HTML-страницы.

Например: #Framework.

<http://www.fio.ru> – адресует на web-сервер страницу по умолчанию (index.html)

[https://(2001:db8::7)Avorkroom/rux/list2.html - адресует страницу list2.html](https://(2001:db8::7)Avorkroom/rux/list2.html%20-%20адресует%20страницу%20list2.html)

<http://213.128.193.13/imgae/blank.gif>

<http://www.global.net:8081/training/generic.asp?pageid=1078&country=DACH> – адресует страницу, динамически создаваемую сервером по запросу

World Wide Web

WWW (World Wide Web) – Всемирная паутина

Гипертекст – совокупность документов, содержащих текстовую, аудио и видеоинформацию, связанных между собой ссылками в единое целое.

Сетевая веб-служба или сервис WWW – рспределенная информационная система

Основной ресурс WWW – веб-сайты

Тимати Джон Бернерс-Ли (изобретатель WWW, создал первый сайт, первый браузер, первый веб-сервер. Придумал http)

Программное обеспечение сервиса WWW

Http-сервер – это программа, которая слушает входящие HTTP-запросы, обрабатывает их и отправляет ответы клиентам.

HTTP-клиент (веб-клиент, браузер) – это программа, которая устанавливает связь с web-сервером по назначенному номеру порта, отправляет запрос и получает ответ.

Веб-серверы

Apache, Internet Information Services, Google Web Server, nginx

Протокол HTTP

HTTP (Hypertext Transfer Protocol) – протокол передачи гипертекста

Протокол прикладного уровня

Работает поверх TCP-протокола

Единица данных – http-сообщение

Использует URI-схемы “http” и “https”. По умолчанию за протоколом HTTP на сервере закреплен порт 80 для схемы “http” и 443 для “https”

Ориентирован на прием сообщений

Изначально – текстовый режим работы, http/2 – бинарный.

HTTP/0.9 – разработана в 1991 году Т. Бернерсом-Ли

HTTP/1.0 (1996) описана RFC 1945

HTTP/1.1 принята в 1999г. Описана в RFC 2616. Добавлено постоянное соединение (режим Keep-alive). Добавлен заголовок Host

HTTP/2 принята в 2015 году. Описана в RFC 7540. Основана на протоколе SPDY, разработанном Google.

HTTP/3 – в стадии разработки и стандартизации

Структура http-сообщения

Начальная строка

<название заголовка>: <значение><\n>

<название заголовка>: <значение><\n>

<\n>

<тело сообщения>

Структура начальной строки запроса клиента:

<метод> <имя запрашиваемого ресурса> <http/версия><\n>

Пример HTTP-запроса:

GET /courses/networks HTTP/1.1

Host: [www.intuit.ru](http://www.intuit.ru)

User-Agent: Mozilla/5.0 (Ubuntu; X11 Linux x86\_64) Gecko/20100

Accept: text/html, image/gif, image/jpeg “/”

Структура начальной строки ответа сервер

HTTP/версия <код состояния> <описание><\n>

Пример HTTP-ответа

HTTP/1.1 200 Ok

Date: FR 10 Jan 2010 07:34:28 GMT

Server: Apache/2.2.6 (Unix)

Content-type: text/html

Content-length: 2482

<html><body

<p>HTML<\p>

<\body><\html>

Методы

Метод – это HTTP-команда с которой начинается первая трока запроса и которая сообщает серверу цели запроса

Основные методы: GET, HEAD, POST, OPTIONS, PUT, DELETE, TRACE

Метод GET

Используется для запроса информации, расположение которой на сервере определяется заданным URI

С помощью метода GET можно начать выполнение процесса на сервере, например, поиск в БД

Клиент может передавать параметры выполнения запроса URI после символа ?

Метод HEAD

Метод HEAD запрашивает только заголовочную информацию по файлу или ресурсу.

Функционально аналогичен методу GET, но сервер ничего не помещает в область тела сообщения.

Метод используется, если клиенту нужна информафия о документе, но не нужно получать сам документ. Например для проверки наличия ресурса (Валидация URI).

Метод POST

Позволяет в клиентском запросе отправить данные серверу.

Передаваемые данные обычно включаются в тело запроса и на сервере передаются программе обработки данных, указанной в URI

Метод POST может использоваться для передачи: входных данных сетевым службам, ключевых слов для поисковой машины, комментариев, файлов и т.д.

Метод PUT

Используется лля передачи на сервер документа, включенного в текст запроса. Документ, отправленный методом PUT, должен быть загружен на указанный URI.

Если по заданному URI не существует ресура, то сервер создает и позвращает статус 201 (Created) с указанием URI нового ресурса в загововке Location. Если ресурс был изменен, то сервер возвращает Ok/

Различие методов POST и PUT. Метод POST, предполагает, что по указанному URI будет производиться обработка данных.

Метод OPTIONS

Запрашивает информацию о поддержке HTTP на web-сервере.

Используется для определения возможностей веб-сервера, поддерживаемый методов, параметров соединения для конкретного ресурса.

Запрос клиента может содержать тех.сообщения для указания интересующих его сведений.

Чтобы узнать возможности сервера, клиент должен указать в URI звезду «\*»

Методы HTTP

DELETE – запрашивает удаление с сервера указанного ресурса

PATCH – аналогичен PUT, но применяется только к фрагменту ресурса

TRACE – используется для трассировки ресурса

LINK – устанавливает связь указанного ресурса с другим

UNLINK – убирает связь указанного ресурса с другим

Каждый сервер обязан поддерживать как минимум методы GET и HEAD

Если сервер не распознал указанный клиентом метод, он должен вернуть код 501 (Not implemented)

Если метод не применим к конкретному ресурсу, то возвращается код 405 (Method not allowed) В обоих случаях сервер должен включить в сообщение ответа заголовок Allow со списков поддерживаемых методов.

HTTP-заголовки

General Headers (Основные заголовки) – могут включаться в любое сообщение клиента и сервера

Request Headers (заголовки запроса) – используется только в запросах клиента.

Response Headers (заголовки ответа) – используется только в ответах сервера

Entity Headers (заголовки сущности) – сопровождают каждую сущность сообщения

Заголовок запроса, ответа и основные заголовоки описывают все сообщение в целом и различаются только в начальном блоке заголовков.

Сущность (entity) – это различные виды передаваемых объектов.

Сущности используются при передаче множественного содержимого (multipart/\*)

Сущность состоит из метаинформации (заголовок сущности) и содержания (тело сущности).

Заголовки сущности характеризуют содержимое каждой сущности и располагаются непосредственно перед ее телом.

Основные заголовки

Date – дата и время создания сообщения

Connection

В HTTP/1.0 – задает постоянное соединение. Введен в версии 1.

Заголовки запроса

Accept – Сообщает серверу типы поддерживаемого контента.

Host – Имя сайта, DNS-имя или IP-адрес и порт компьютера, на котором расположен сервер. Введен в версии 1.1. Используется, когда на сервере расположено несколько сайтов.

User-Agent – информация о клиенте (обычно имя браузера)

Заголовки ответа

Location – определяет точное расположение ресурса к которому может быть перенаправлен клиент

Server – содержит информацию о ПО сервера

Заголовки сущностей

Content-type

Content-Length

Expire

Last-Modified

Content-Encoding

Content-Language

Allow

Код состояния

1xx – информация о процессе передачи

2xx – информация об успешном принятии и обработке запроса клиента

3xx – для успешного выполнения операции нужно произвести следующий запрос по URI-адреса, указанному в дополнительном заголовке Location

4xx – информация об ошибках на стороне клиента.

5xx – ошибки по вине сервера