Третья лекция
java for web
HTTP

Hypertext Transfer Protocol

Сетевой протокол — набор правил и действий (очерёдности действий), позволяющий осуществлять соединение и обмен данными между двумя и более включёнными в сеть устройствами.

Широко распространённый протокол передачи данных, предназначенный для передачи гипертекстовых документов (то есть документов, которые могут содержать ссылки, позволяющие организовать переход к другим документам).

Аббревиатура HTTP расшифровывается как HyperText Transfer Protocol, «протокол передачи гипертекста».

Когда вы заходите в браузер, не важно, какой именно браузер у вас установлен, и вводите в адресную строку адрес к сайту, то браузер автоматически прибавляет к адресу приставку «http://». Единственное, эта приставка может быть по умолчанию скрыта, но если скопировать адрес и вставить его в другое место, то ее без труда можно будет увидеть.

Эта приставка обозначает, что вы будете обращаться к ресурсу по протоколу HTTP.

Свойства протокола HTTP

Протокол HTTP предполагает использование клиент-серверной структуры передачи данных. Клиентское приложение формирует запрос и отправляет его на сервер, после чего серверное программное обеспечение обрабатывает данный запрос, формирует ответ и передаёт его обратно клиенту. После этого клиентское приложение может продолжить отправлять другие запросы, которые будут обработаны аналогичным образом

Задача, которая традиционно решается с помощью протокола HTTP — обмен данными между пользовательским приложением, осуществляющим доступ к веб-ресурсам (обычно это веб-браузер) и веб-сервером. На данный момент именно благодаря протоколу HTTP обеспечивается работа интернета.

Конечно же, он используется не только для передачи HTML-файлов, но и для любых других объектов: картинок, скриптов, CSS-файлов, файлов данных. Также он работает и в обратную сторону — для заливки на сервер файлов, отправки форм и т. п. AJAX-приложения также, общаются с сервером по HTTP.

Как правило, передача данных по протоколу HTTP осуществляется через TCP/IP-соединения. Серверное программное обеспечение при этом обычно использует TCP-порт 80 (и, если порт не указан явно, то обычно клиентское программное обеспечение по умолчанию использует именно 80-й порт для открываемых HTTP-соединений), хотя может использовать и любой другой.

Transmission Control Protocol/Internet Protocol

Каждый компьютер (он же: узел, хост) в рамках сети Интернет тоже имеет уникальный адрес, который называется IP-адрес (Internet Protocol Address), например: 195.34.32.116.

IP адрес состоит из четырех десятичных чисел (от 0 до 255), разделенных точкой. Но знать только IP адрес компьютера еще недостаточно, т.к. в конечном счете обмениваются информацией не компьютеры сами по себе, а приложения, работающие на них. А на компьютере может одновременно работать сразу несколько приложений (например почтовый сервер, веб-сервер и пр.).

Большинство серверных приложений имеют стандартные номера, например: почтовый сервис привязан к порту с номером 25 (еще говорят: «слушает» порт, принимает на него сообщения), веб-сервис привязан к порту 80, FTP - к порту 21 и так далее.

В компьютерных сетях, работающих по протоколам TCP/IP, аналогом бумажного письма в конверте является пакет, который содержит собственно передаваемые данные и адресную информацию — адрес отправителя и адрес получателя.

Комбинация: "IP адрес и номер порта" - называется "сокет".

Ввзаимодействие осуществляется по схеме «клиент-сервер»: "клиент" запрашивает какую-либо информацию (например страницу сайта), сервер принимает запрос, обрабатывает его и посылает результат.

Любой цифровой IP адрес можно связать с буквенно-цифровым именем, преобразованием доменного имени в цифровой IP адрес занимается сервис доменных имен — DNS (Domain Name System).

DNS серверу отправляется запрос (точнее пакет с запросом) на сокет 195.34.32.116:53. Как было рассмотренно выше, порт 53 соответствует DNS-серверу - приложению, занимающемуся распознаванием имен. А DNS-сервер, обработав наш запрос, возвращает IP-адрес компьютера, который соответствует введенному имени.

Далее наш компьютер устанавливает соединение с портом 80 компьютера и посылает запрос (пакет с запросом) на получение страницы. 80-й порт соответствует веб-серверу. В адресной строке браузера 80-й порт как правило не пишется, т.к. используется по умолчанию, но его можно и явно указать после двоеточия.

Приняв от нас запрос, веб-сервер обрабатывает его и в нескольких пакетах посылает нам страницу в на языке HTML - языке разметки текста, который понимает браузер.

Наш браузер, получив страницу, отображает ее. В результате мы видим на экране главную страницу этого сайта.

И так. IP протокол — это протокол так называемого сетевого уровня. Задача этого уровня — доставка ip-пакетов от компьютера отправителя к компьютеру получателю.

TCP — это протоколытак называемого транспортного уровня. Транспортный уровень находится над сетевым. На этом уровне к пакету добавляется порт отправителя и порт получателя.

этот протокол с установлением соединения и с гарантированной доставкой пакетов. Сначала производится обмен специальными пакетами для установления соединения, происходит что-то вроде рукопожатия (-Привет. -Привет. -Поболтаем? -Давай.). Далее по этому соединению туда и обратно посылаются пакеты (идет беседа), причем с проверкой, дошел ли пакет до получателя. Если пакет не дошел, то он посылается повторно («повтори, не расслышал»).

Схема взаимодействия по протоколу HTTP

1 этап. Клиент (браузер) отправляют строку запроса (HTTP-запрос), которая создается по определенным правилам, и запрашивает нужную веб-страничку на сервере.

2 этап. Сервер принимает запрос и ищет у себя эту веб-страницу. По результатам этого поиска создается ответ клиенту (HTTP-ответ). Этот ответ тоже оформляется по определенным правилам.

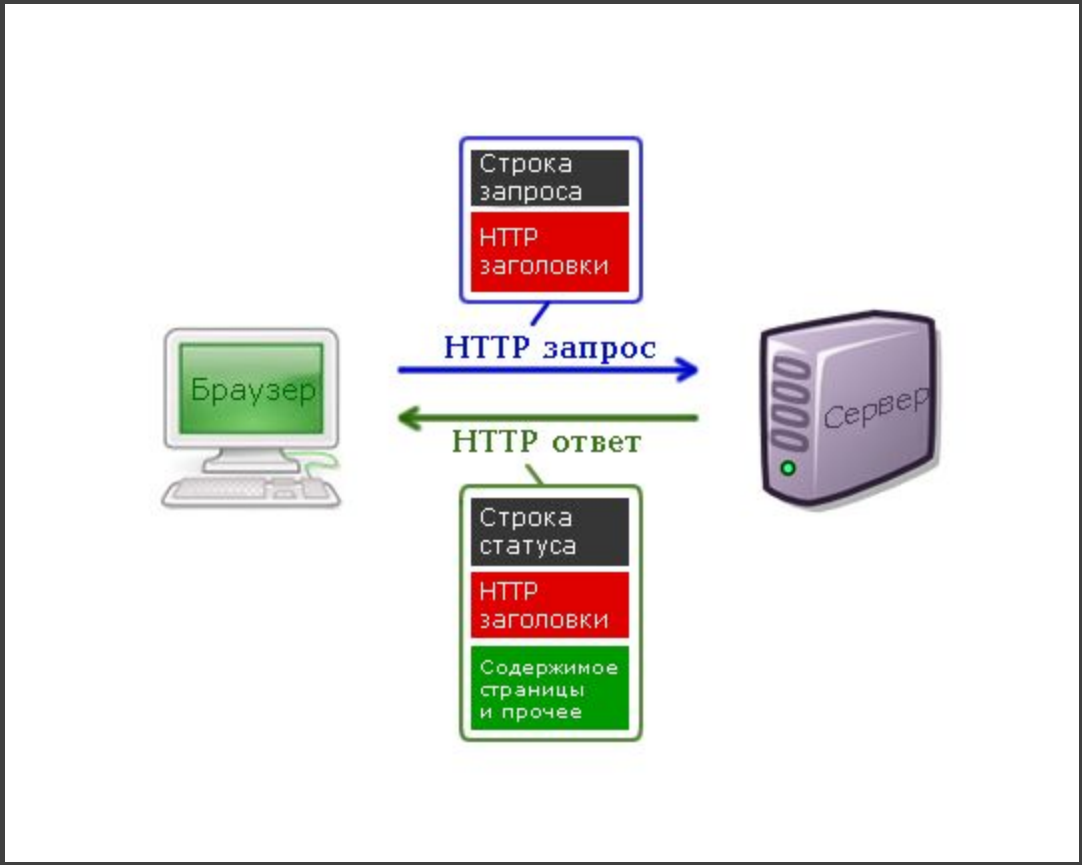
Если все прошло успешно и страница найдена, то в этом ответе будет передан код нужной веб-страницы + дополнительная служебная информация.

Если произошел, какой-то сбой, то будет передан код ошибки и дополнительная служебная информация.

В отличие от многих других протоколов, HTTP не сохраняет своего состояния. Это означает отсутствие сохранения промежуточного состояния между парами «запрос-ответ».

Компоненты, использующие HTTP, могут самостоятельно осуществлять сохранение информации о состоянии, связанной с последними запросами и ответами (например, «куки» на стороне клиента, «сессии» на стороне сервера).

Схема взаимодействия по протоколу HTTP



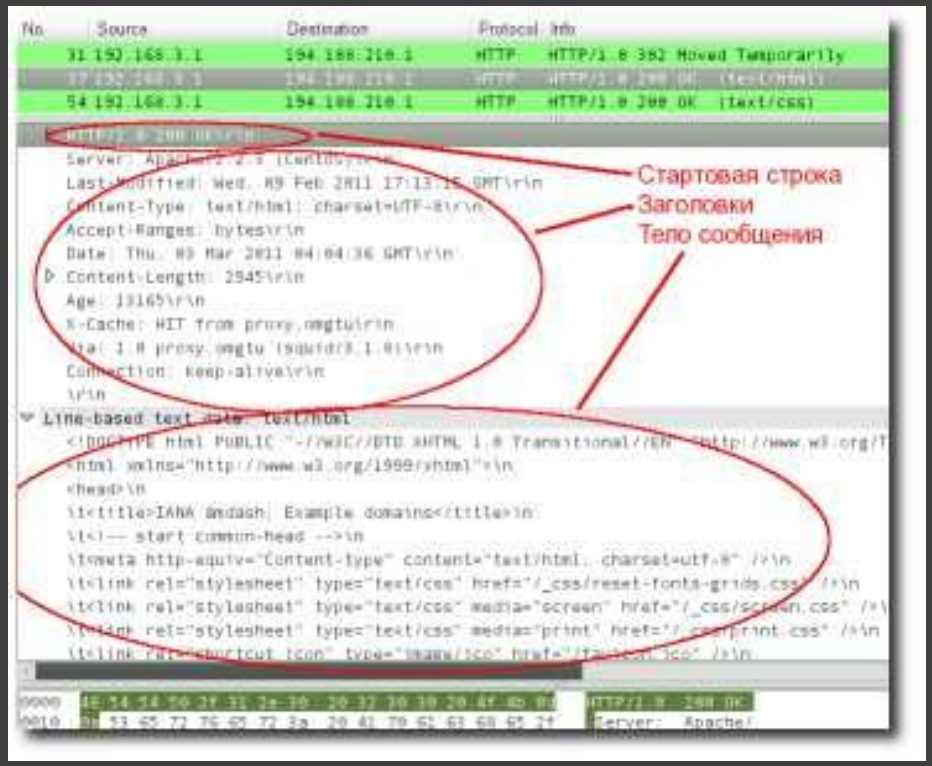
Структура протокола

Структура протокола определяет, что каждое HTTP-сообщение состоит из трёх частей, которые передаются в следующем порядке:

Стартовая строка — в ней указывается тип сообщения.

Заголовки — описывают тело сообщения, определяя его параметры.

Тело сообщения — само сообщение, должно отделяться пустой строкой.



Стартовая строка HTTP

Стартовая строка является обязательным элементом, так как указывает на тип запроса/ответа, заголовки и тело сообщения могут отсутствовать.

Стартовые строки различаются для запроса и ответа. Строка запроса выглядит так:

Метод URI HTTP/Версия протокола

Пример запроса:

GET /web-programming/index.html HTTP/1.1

Стартовая строка ответа сервера имеет следующий формат:

HTTP/Версия КодСостояния [Пояснение]

Например, на предыдущий наш запрос клиентом данной страницы сервер ответил строкой:

HTTP/1.1 200 Ok

HTTP Header

Заголовок HTTP (HTTP Header) — это строка в HTTP-сообщении, содержащая разделённую двоеточием пару вида «параметр-значение».

Как правило, браузер и веб-сервер включают в сообщения более чем по одному заголовку.

Каждый запрос имеет как минимум заголовок, который сообщить серверу информацию о своей конфигурации и данные о форматах документов, которые он может принимать.

Каждый ответ состоит из заголовка ответа (информация о сервере и передаваемых данных).

Заголовки делятся на 4 группы:

Основные заголовки — обязательно включаются в любое сообщение клиента и сервера.

Заголовки запроса — можно встретить только в запросах от клиента.

Заголовки ответа — можно встретить только в ответах от сервера.

Заголовки запроса и ответа, как и основные заголовки, описывают всё сообщение в целом и размещаются только в начальном блоке заголовков,

Заголовки сущности — описывают сущность каждого сообщения (может относиться как к клиенту, так и к серверу).

В отдельный класс заголовки сущности выделены, чтобы не путать их с заголовками запроса или заголовками ответа при передаче множественного содержимого (multipart/*). Заголовки сущности характеризуют содержимое каждой части в отдельности, располагаясь непосредственно перед её телом.

Заголовок Accept

Заголовок Accept предназначен для информирования сервера о типах данных, которые поддерживаются клиентом (браузером). В этом заголовке браузер перечисляет, какие типы документов он "понимает". Перечисление идет через запятую.

Accept: text/html, text/plain, image/jpeg

В последнее время вместо списка указывается значение *.*, что означает "все типы".

Заголовок Content-type

Данный заголовок предназначен для идентификации типа передаваемых данных.

Обычно для этого заголовка указывается значение application/x-www-form-urlencoded. Сервер никак не интерпретирует рассматриваемый заголовок, а просто передает его сценарию через переменную окружения.

Пример: Content-type: text/plain

Заголовок Content-length

Этот заголовок содержит строку, в которой записана длина передаваемых данных в байтах при использовании метода передачи POST.

Заголовок Pragma

Данный заголовок используется для различных целей, одна из которых - это запрет кэширования документа.

Пример заголовка: Pragma: no-cache

Заголовок User-Agent

Содержит версию браузера. Например: User-Agent: Mozilla/5.0 (compatible; Konqueror/3.0.0-10; Linux).

Accept-Language

Сообщает допустимые языки содержания и их приоритет, именно от него зависит язык отображения сайта.

Referer

Сообщает о странице, с которой пришел пользователь. Заголовок, сильно полезный веб-мастерам для отслеживания путей попадания на их сайты.

Accept-Charset

Сообщает допустимые кодировки и их приоритет.

X-Requested-With

Нестандартный заголовок, сообщает средство запроса. Используется при запросах из JavaScript без перезагрузки страницы.

Request Headers (Заголовки запроса)

POST http://www.vdata.de/vdata-rechner/av_rente.jsp HTTP/1.0
Accept: image/gif, image/x-xbitmap, image/jpeg, */*
Referer: http://www.vdata.de/vdata-rechner/av_rente.jsp
Accept-Language: de
Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0;)
Host: www.vdata.de
Content-Length: 179
Pragma: no-cache
Cookie: JSESSIONID=42DFB2744509A9E0AC7A556F3BF288F9

submitted=true&geburtsjahr=1945&geschlecht=0&familienstand=0

Response Headers (Заголовки ответа)

HTTP/1.1 200 OK
Server: Sun-ONE-Web-Server/6.1
Date: Fri, 19 Nov 2004 19:18:35 GMT
Content-type: text/html;charset=ISO-8859-1
Set-cookie:
JSESSIONID=5B0A8F3BADFD718A5D55DBEECFDFC
F99;Path=/
Connection: close

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
...

GET vs. POST

Метод GET - Пожалуй самый важный метод. Он используется для запроса данных с ресурса, если эти данные имеют URL. Так же с помощью этого метода начинаются какие либо действия.

Клиент может передавать параметры запроса в самом URI, после символа «?»:

Метод POST занимается отправкой данных, находящихся в теле сообщения, на сервер. Также часто используется для отправки информации из веб-форм и файлов. Простейший пример — написание поста на форуме или комментария в соцсети. После того, как вы написали ваше сообщение, браузер формирует POST-запрос и в его тело помещает ваше сообщение,а затем отправляет его на сервер.

GET – параметры запроса (данные формы) в строке запроса. Проблемы с длиной строки:

```
GET http://192.168.0.5:8080/dms/ma_input1.jsp?
    roomType=1&ambulance=0 HTTP/1.0
```

POST - параметры запроса в его теле – нет ограничений на количество и длину

Коды состояния

Код состояния информирует клиента о результатах выполнения запроса и определяет его дальнейшее поведение. Набор кодов состояния является стандартом, и все они описаны в соответствующих документах RFC.

Каждый код представляется целым трехзначным числом. Первая цифра указывает на класс состояния, последующие - порядковый номер состояния . За кодом ответа обычно следует краткое описание на английском языке.

1xx – информационные

2xx - успешные

200 OK

3xx – перенаправление

301 – Moved Permanently

4xx - ошибки клиента

400 Bad Request

401 Unauthorized

403 Forbidden

404 Not Found

5xx – ошибки сервера

500 Internal Server Error

Установка параметров запроса

При помощи HTML форм

- Установка значения элементов формы
- Submit формы
  - Пользователем в браузере
  - JavaScript на странице

В строке запроса GET:

http://localhost/my.jsp?val1=AAA&val2=999

Установка параметров запроса пользователем

```
<form method="post" action="page2.jsp">
<input type="text" name="hello">
...
<input type="submit" value="next">
</form>
```

•Установка параметров запроса при помощи JavaScript

```
<form method="post" name="mainform" action="my.jsp">
<input type="text" name="hello">
</form>

<a href="javascript:next()">Next</a>
....
<script>
  function next(){
    document.mainform.submitted.value = true;
    document.mainform.submit();
  }
</script>
```