Юнев Владимир Александрович
Эксперт по стратегическим технологиям, Microsoft.

Как работают веб-серверы?

В №5 нашего журнала за 2013 г. была опубликована статья «Как устроен и работает Интернет», в которой рассматривались вопросы происхождения Интернета и базовые технологии сети. В связи с большим интересом читателей мы решили продолжить эту тему и предлагаем вам новый материал, связанный с работой «скрытых» тружеников Интернета – веб-серверов.

Важнейшими изобретениями, ставшими основой современной глобальной сети, являются технологии гипертекста, языка разметки HTML, передачи гипертекста с протоколом http и появление первых интернет-браузеров.

Огромный вклад в изобретение и создание этих технологий внёс Тим Бернерс-Ли (Tim Berners-Lee) – британский учёный, который работал над концепцией технологий Интернета в конце 80-х годов в CERN (Европейская организация по ядерным исследованиям).

Рис. 1. Фотография Тима Бернерса-Ли. Источник – Wikipedia.com

Бернерс-Ли предложил глобальный гипертекстовый проект, в котором содержимое (контент) страниц, состоящее из информации разного типа – текста, изображений, других медиа-форматов – соединялось бы друг с другом и само с собой посредством **гиперссылок**. Сегодня, когда мы щёлкаем в веб-браузере на ссылку, мы не задумываемся над тем, как такая простая концепция появилась в реальной жизни. Кажется, что идея гиперссылок лежит на поверхности, однако, как и все подобные идеи, она потребовала времени на изобретение и воплощение в том практическом виде, в котором мы сейчас её используем. Проект объединения документов в сети с помощью гипертекста был назван Бернерсом-Ли «Всемирной паутиной» (Word Wide Web).

Контент с гиперссылками потребовал новый формат представления информации, который был разработан в виде спецификации **гипертекстового языка разметки,** или **HTML** (Hyper Text Markup Language). По-

требовалось изобрести и правила (или протоколы) обмена гипертекстовыми данными между компьютерами в сети. Таким протоколом стал стандарт **протокола передачи гипертекста**, или http (Hyper Text Transfer Protocol).

Бернерс-Ли жив и сегодня, он продолжает свою работу над всемирной паутиной в качестве главы специальной независимой организации W3C (консорциум всемирной паутины), которая разрабатывает и утверждает спецификации и стандарты технологий, используемых в глобальной сети.

«Сервер – это любой компьютер, на котором запущено программное обеспечение, отвечающее на запросы на получение документов и других данных. Программы, которые запрашивают и отображают документы (такие, как браузер), называются клиентами. Термины «серверный» и «клиентский» применительно к функциям обозначают, на какой машине производится обработка. Клиентские функции выполняются на машине пользователя, серверные функции – на удалённой машине.»

Д.Н. Роббинс «WEB-дизайн»

Протокол HTTP и запросы

Важнейшей частью современного Интернета является протокол обмена гипертекстовыми данными HTTP (Hyper Text Transfer Protocol). Этот протокол обладает массой положительных качеств, во главе которых стоит простота.

Вообще, протокол – это правила обмена данными между двумя разными точками. Такими точками в Интернете являются ваш компьютер и компьютер в сети, с которого вы хотите получить информацию, например, веб-сайт microsoft.com, который содержит новости, ссылки на программы и другую информацию, представленную в гипертекстовом виде.

Протокол HTTP построен по принципу запрос-ответ. Клиентский компьютер посылает запрос в определённом формате (как правило, это специально оформленная строка текста), на что сервер или отвечающий компьютер в сети должен вернуть определённый ответ, понятный клиенту.

Каждый такой запрос клиента на сервер (и ответ сервера) состоит из трёх частей:

1) стартовая строка,
2) заголовки,
3) тело сообщения.

Рассмотрим конкретный пример запроса. В ответ на ваше желание посетить сайт http://microsoft.com ваш браузер отправит запрос на сервер (компьютер в сети) примерно следующего содержания:

```
GET /en/us/default.aspx HTTP/1.1
Host  www.microsoft.com
Accept text/html, application/xhtml+xml, */*
Accept-Language  ru-RU
User-Agent  Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1;
WOW64; Trident/5.0)
Accept-Encoding  gzip, deflate
```

Здесь клиент (браузер, которым вы пользуетесь) сформировал 6 строк: первая стартовая строка и пять строк заголовков. Стартовая строка очень важна при обмене данными, она указывает на тип запроса, адрес необходимого содержимого и протокол по которому клиент собирается принять данные.

Стартовая строка запроса

В приведённом выше примере клиент отправляет на сервер запрос типа **GET** с просьбой вернуть содержимое файла по адресу **/en/us/default.aspx**, используя протокол передачи данных HTTP версии 1.1.

Что такое «запрос типа GET»? Запросы на сервер могут быть нескольких типов, тип определяется в стартовой строке. Перечислим самые значимые типы запросов: GET, POST, HEAD, PUT, DELETE. Назначение запроса сервер понимает из его типа. Например, запрос типа GET (получить) означает, что клиент просит передать ему определённое содержимое на сервере.

Другие запросы означают для сервера следующие действия:

- **POST** (отправить) – клиент отправляет с запросом некие данные, возможно, заполненную форму, которые необходимо обработать;
- **HEAD** (заголовок) – клиент просит вернуть ему одни только заголовки ответа, но не содержимое ресурса. Этот тип запроса используется для получения метаданных (сопутствующих данных) ресурса, которые могут использоваться вашим браузером для дальнейшей работы;
- **PUT** (поместить) – этот тип запроса похож на POST, но используется с целью добавить данные на удалённый ресурс;
- **DELETE** (удалить) – запрос с таким типом запроса предполагает удаление некоего содержимого на сервере.

Существует ещё ряд типов запросов, но их использование не так распространено.

Кроме типа запроса и адреса содержимого, которое мы запрашиваем с сервера, стартовая строка содержит ещё один параметр – протокол передачи данных и его версию. В нашем случае этот параметр содержит значение HTTP/1.1.

Протокол http за свою долгую историю изменялся крайне редко и, это может показаться удивительным, имеет всего три версии: 0.9, 1.0 и 1.1. Объяснить такую стабильность протокола можно его простотой, заложенной изначально. Кроме того, такая фундаментальная вещь, как правила обмена информацией в глобальной сети, не должна меняться часто. И всё же существуют разные версии протоколов HTTP, которые содержат минимальные различия между собой. Именно поэтому клиент и сервер обмениваются данными о версиях протоколов, которые они поддерживают. Это позволяет найти клиентскому компьютеру и серверу в сети «общий язык», который понятен обоим и на котором они могут разговаривать.

Заголовки запроса

Последующие данные запроса – это заголовки, которые так же очень важны при обмене данными. Заголовки, в отличие от стартовой строки, могут иметь произвольные данные формата ключ-значение. В на-

шем случае **Host, Accept, Accept-Language, User-Agent и Accept-Encoding** – это ключи, а данные напротив – это значения ключей.

Для чего нужны заголовки? Заголовки – это возможность передать на сервер дополнительную информацию, которая корректирует запрос или добавляет в него условия. Заголовки могут рассказывать о клиенте серверу, чтобы тот предоставлял пользователю более точные данные.

Например, заголовок Accept-Language, который автоматически формирует браузер, рассказывает серверу о том, какой предпочтительный язык используется на компьютере пользователя (в нашем случае, ru-RU – это русский). Имея такие данные от клиента, сервер может автоматически вернуть вместо англоязычной страницы русскоязычную страницу. А это, согласитесь, уже большая помощь пользователю.

Другие заголовки не менее важны и полезны:

- **Host** – определяет относительно какого имени компьютера необходимо вернуть содержимое, в нашем случае мы запрашиваем содержимое **/en/us/default.aspx с сервера www.microsoft.com.**
- **Accept** – определяет формат содержимого (контента), в котором мы ожидаем увидеть запрошенные данные. В нашем случае мы просим вернуть данные в формате **text/html, application/xhtml+xml, */*.** Эти форматы данных понятны браузеру. Получив данные в таких форматах, браузер сможет отобразить их в понятном пользователю виде.
- **User-Agent** – содержит строку, в которой передаётся информация о вашем браузере. Эта информация важна серверу, поскольку из-за большого разнообразия браузеров иной раз контент, возвращаемый пользователю, требуется видоизменить. Особенно полезно бывает изменить контент, когда пользователь заходит на сайт с мобильного устройства (смартфона).
- **Accept-Encoding** – строка, которая говорит серверу о том, что клиент поддерживает возможность сжатия данных при передаче по определённым алгоритмам (**gzip, deflate**). Сжатие данных при передаче позволяет быстрее передать данные, и вы быстрее получите ответ (веб-страницу) от сервера. В нашем случае браузер сообщает серверу о том, что он готов принять данные в сжатом виде, а сервер, если он обладает функцией сжатия, теперь может вернуть сжатые данные.

Существует огромная масса других заголовков, которые могут быть переданы от клиента на сервер. Вы можете определять собственные заголовки, если уверены, что сервер сможет распознать их.

Ответ сервера и содержимое запроса (тело сообщения)

Последней частью сообщений, передаваемых между клиентом и сервером, является тело сообщения, или его содержимое. В случае GET-запроса тело сообщения может быть пустым, так как мы производим запрос и не передаём данные. Когда клиент формирует запросы другого типа, например POST или PUT, то тело сообщения содержит данные, которые клиент хочет отправить на сервер.

В ответ на наш GET-запрос сервер формирует данные, например, генерирует веб-страницу и возвращает её нам вместе с сообщением, которое имеет тот же формат, что и запрос:

HTTP/1.1 200 OK
Cache-Control public
Content-Type text/html; charset=utf-8
Expires Mon, 14 Mar 2011 10:41:48 GMT
Last-Modified Mon, 14 Mar 2011 07:01:03 GMT
ETag 634356576630000000
Server Microsoft-IIS/7.5
X-AspNet-Version 4.0.30319

Наверняка, вы уже распознали структуру этих данных. Первой строкой идёт стартовая строка, а затем заголовки сообщения. Вслед за этими данным, как правило, следует содержимое, которое мы запрашивали. В нашем случае это веб-страница /en/us/default.aspx. На рисунке 2 представлено это содержимое, отображённое с помощью инструментов разработчика в браузере Internet Explorer 9.

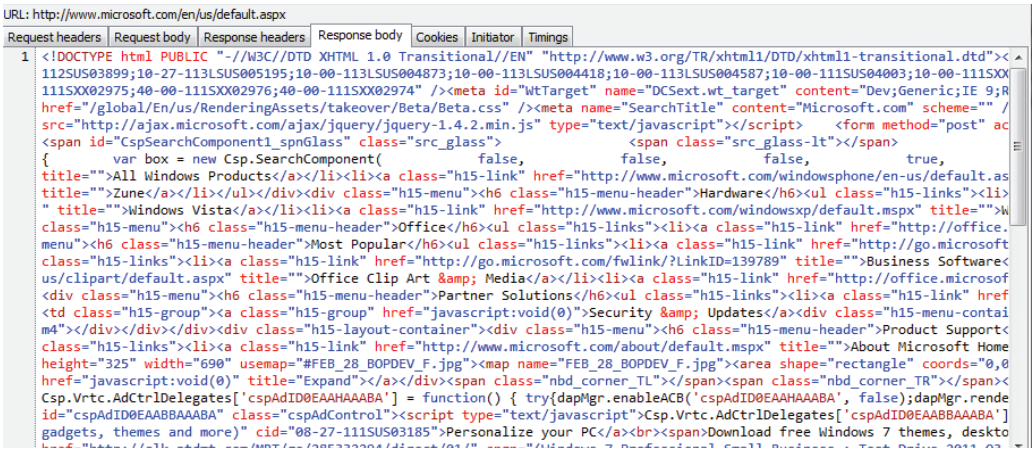

Рис. 2. Содержимое тела сообщения, полученное в ответ на запрос

Получив от сервера содержимое, браузер отображает его в понятном пользователю виде, например, в виде веб-страницы.

Вернёмся к ответу сервера. Обратите внимание на стартовую строку: **HTTP/1.1 200 OK**. Вы можете определить, что первым параметром стартовой строки в ответе сервера идёт протокол и его номер. Таким образом, сервер в ответ на запрос сообщает клиенту, что он готов «разговаривать» на языке протокола HTTP версии 1.1.

Следующим за протоколом параметром является код состояния (результата запроса). Это очень важный параметр и свойство протокола HTTP. В ответ на запросы клиента сервер должен уметь сообщать о своём состоянии, о возможных ошибках при запросе, о статусе выполняемой операции. Таким сообщением для клиента являются коды состояния. В нашем случае, сервер возвращает код **200**. Это означает, что запрос завершился без ошибок. Однако, существует большое количество других кодов статуса:

- **301** – код сообщает клиенту, что ресурс, который он запросил, был перемещён на новое место. Обычно в ответ на это клиент снова запрашивает содержимое, но уже с нового местоположения. Такой процесс называется редиректом (пере-

направлением);

- **304** – код сообщает клиенту, что содержимое, которое он запросил, не изменялось со времени последнего запроса. В связи с этим клиент может использовать данные, которые он уже загрузил для повторного отображения, это увеличивает скорость отображения страниц и исключает ненужные запросы на сервер;
- **401** – код сообщает клиенту, что запрос требует идентификации, то есть передачи логина и пароля;
- **403** – код сообщает клиенту, что такой запрос запрещён в связи с ограничениями прав пользователя;
- **404** – наверное, наиболее известный в Интернете код, который говорит клиенту о том, что ресурс, который он пытается запросить (например, веб-страница), на сервере отсутствует (страницу могли удалить или пользователь ошибся при вводе адреса);
- **500** – код сообщает клиенту, что при обработке запроса на сервере произошла ошибка и данные не могут быть возвращены. Такого рода коды показывают, что на сервере содержатся какие-то ошибки или на нём ведутся администраторские работы.

Существует огромное множество других кодов, которые используются реже. Полный перечень можно посмотреть в Wikipedia.

Следующий за кодом статуса текст является сопроводительным к статусу. В нашем случае для кода 200 он имеет значение OK, то есть «всё хорошо». Для кода с номером 304 такой текст будет иметь значение «Notmodified», то есть «не было изменено».

Обратите внимание на заголовки сообщения, которые нам вернул сервер. Естественно, что они отличаются от тех заголовков, которые передавал на сервер наш браузер. Давайте рассмотрим их подробнее.

Несколько заголовков Cache-Control, Expires, Last-Modified и ETag отвечают за состояние контента, который был возвращён сервером. Заголовок Last-Modified содержит дату последнего изменения контента на сервере. Expires сообщает клиенту, когда контент будет считаться устаревшим. Все эти данные предназначены для повышения эффективности обмена данными через механизм кэширования. Кэширование – это процесс сохранения некоторого контента для повторного использования без загрузки с сервера. Если некоторое содержимое находится в кэше вашего браузера, то браузер будет стараться использовать его тогда, когда это возможно. В определении возможности использования контента из кэша браузеру помогают заголовки сервера, с которыми контент был получен. Например, основываясь на Expires, браузер может знать, когда следует запросить контент повторно.

Заголовок Server со значением Microsoft-IIS/7.5 сообщает клиенту о том, какой веб-сервер используется на сервере. В нашем случае таким веб-сервером является сервер IIS версии 7.5.

Последний из рассматриваемых заголовков, X-AspNet-Version, является нестандартным, то есть введённым компанией Microsoft самостоятельно для того, чтобы информировать клиента о том, какая версия платформы веб-разработки ASP.NET установлена на сервере. Эти сведения могут быть полезны при разработке разного программного обеспечения.

Юмор Юмор Юмор Юмор Юмор Юмор

– Что вы так волнуетесь? Боитесь моих вопросов?
– Да нет, боюсь своих ответов.