ЛИТЕРАТУРА

1. Коверзин, Д. Особенности охранных систем [Электронный ресурс] / Д. Коверзин. – Режим доступа: http://sigadoma.ru/oxrannaya-signalizaciya/istoriya-razvitiya-oxrannoj-signalizacii.html. – Дата доступа: 10.03.2015.
2. Охранные системы [Электронный ресурс]. – Режим доступа: http://www.sob.by/safe.php. – Дата доступа: 12.04.2015.
3. ОАО ”АЛАРМ”, a6_operating_manual_part 1/ОАО”АЛАРМ” [Электронный ресурс]. – Минск, 2012. – Режим доступа: http://www.rovalant.com/download/a6_operating_manual_part1.pdf-c.19-25, 39-45.
4. SimCom, datasheet Sim 900 [Electronic resource]. – 2012. – Mode of access : http://amigalounge.com/files/SIM900_HD_V1.01(091226).pdf. – Date of access : 04.05.2015.
5. Дмитренко, Д. Прибор сигнализации GSM на основе модуля SIM900 [Электронный ресурс] / Д. Дмитренко. – 2009. – http://ddn.radioliga.com/cnt/11.htm. – Дата доступа: 03.03.2015.
6. Нестеров, В. Вопросы безопасности передачи данных [Электронный ресурс] / В. Нестеров, О. Пушкарев. – 2010. – Режим доступа: http://www.wireless-e.ru/assets/files/pdf/2008_4_32.pdf. – Дата доступа: 19.03.2015.
7. Робль, Р. Прогрессивные GSM-модули на J2ME [Электронный ресурс] / Р. Робль, З. Марич. – 2013. –Режим доступа: http://www.wireless-e.ru/assets/files/pdf/2010_02_12.pdf. – Дата доступа: 25.03.2015.

УДК 004.051

УВЕЛИЧЕНИЕ СКОРОСТИ ЗАГРУЗКИ ВЕБ-САЙТОВ

Ю.В. ЛАПТЕВ
(Представлено: канд. техн. наук, доц. Р.П. БОГУШ)

Рассматриваются наиболее полезные советы по оптимизации скорости загрузки веб-сайтов на стороне клиента.

С каждым годом Интернет растет очень быстро. Увеличивается пропускная способность каналов, а вместе с ними и количество передаваемого трафика. Сайты становятся больше по размеру и сложнее во взаимодействии. Размеры загружаемых файлов увеличиваются многократно, а время ожидания пользователей не уменьшается. За последние 5 лет средний размер веб-страниц вырос в 5 раз, а за последний год – в два раза. При этом каждая страница использует много различных объектов, что негативно сказывается на общем времени загрузки. Только около 5-10% от общего времени загрузки приходится на серверную часть. Все остальное составляет именно клиентская архитектура. Что обычно видит пользователь, заходя на сайт, и как долго он это видит? 70% посетителей уйдут после 10 секунд пребывания на сайте. При этом наиболее характерным временем ожидания будет 4 секунды: если за это время сайт загружается у 90% пользователей, то это считается быстрым веб-ресурсом. Однако многие компании пытаются выжать из своего сайта максимум по скорости работы. Недаром высоконагруженные проекты типа Google, Amazon, Вконтакте, Facebook и Одноклассники очень серьезно подходят к вопросу скорости загрузки своих сайтов. За каждым потерянным моментом времени кроется определенная сумма денег [1].

При помощи различных способов оптимизации загрузки веб-страниц удастся в разы уменьшить время ожидания полной загрузки страницы, а также сохранить больше активных пользователей, особенно среди тех, кто испытывает различные проблемы сетевого соединения.

Основной раздел. Клиентская оптимизация — это оптимизация процесса загрузки клиентским приложением содержимого веб-страниц. Основная цель такого процесса — достижение максимальной скорости загрузки страниц сайта браузером клиента. При построении высокопроизводительных сайтов должен присутствовать и серверный, и клиентский подход, они много в чем дополняют друг друга. Главное отличие клиентского подхода состоит в том, что в качестве объекта оптимизации рассматриваются страницы сайта, состоящие из HTML-документа, содержащего вызовы внешних объектов, а также сами внешние объекты (чаще всего это файлы стилей, скрипты и изображения). Различные технологические решения клиентской области сайта при одинаковой нагрузке на сервер могут обеспечивать совершенно разные характеристики клиентского быстродействия. При исключении из рассмотрения всех факторов, относящихся к серверному программному обеспечению и каналу передачи данных, можно заключить, что увеличение скорости загрузки страницы на различных стадиях загрузки принципиально возможно за счет ограниченного количества методов. Об этих методах и пойдет речь далее[1].

Одним из основных способов по увеличении скорости загрузки является уменьшение количества HTTP-запросов. Около 80% загрузки страницы ориентировано на загрузку компонентов страницы: скриптов, изображений, стилей, flash. Спецификация HTTP/1.1 советует, чтобы браузеры параллельно загружали не более 2-х компонентов веб-страницы с одного хоста. Уменьшив количество этих компонентов мы уменьшаем количество HTTP-запросов к серверу и как результат увеличиваем скорость за-

грузки страницы. Для уменьшения количества запросов используют следующие способы: объединение нескольких файлов в один, использование Inline-картинок или CSS-спрайтов. Однако если на сайте слишком много графической информации, то следует использовать параллельное скачивание. Загружаемый контент выносится на отдельный поддомен. Это может быть один и тот же сервер, но браузер будет считать, что они разные. Чем больше поддоменов будет создано, тем больше файлов браузер сможет одновременно загрузить и тем быстрее загрузится вся страница сайта.

Все подключаемы стили и скрипты должны быть минимизированы. Минимизация файла – это удаление из кода всех несущественных символов с целью уменьшения объема файла и ускорения его загрузки. В минимизированном файле удаляются все комментарии и незначащие пробелы, переносы строк, символы табуляции. Чем меньше объем файла, тем меньше времени понадобится браузеру на его загрузку. Существует множество различных онлайн сервисов которые могут сделать сжатие и оптимизацию стилей и скриптов. Также можно уменьшить размера кода самой страницы, используя способы верстки, которые требуют минимум тегов HTML и правил CSS. Например, семантическая верстка с применением независимых блоков более предпочтительна, чем верстка вложенными таблицами с использованием избыточных тегов.

Многие разработчики производят подключение CSS стилей в конце страницы и это не позволяет многим браузерам рендерить страницу постепенно. Это объясняется тем, что браузер «не хочет» перерисовывать элементы, у которых после загрузки страницы может измениться стиль. Так что все свои файлы стилей всегда подключайте в верхней части страницы в секции HEAD. Помещая подключение к css-файлам в хедере страницы мы получаем постепенную загрузку — сначала заголовок, потом логотип наверху, навигация и другие элементы. Это в свою очередь служит отличным индикатором загрузки страницы для пользователя и улучшает общее впечатление от сайта.

Подключать js-файлы следует в отличие от стилей внизу страницы, что позволит браузеру загрузить страницу с контентом в первую очередь, а уже потом начать загрузку js-файлов. Это особенно важно, если сайт содержит все возможные интерактивные «примочки», то различных скриптов может быть много и весить они могут несколько сотен килобайт, поэтому перед загрузкой страницы заставлять пользователя ждать пока загрузятся все скрипты губительно.

Браузеры и прокси-серверы обычно стремятся сохранить максимум информации в своих хранилищах, для того чтобы ускорить повторную загрузку ранее загруженных объектов. Важно помнить, что при этом возможна потеря актуальности представляемых данных, поэтому политика кэширования должна быть организована с учетом всех возможных ситуаций. Кэширование – это один из наиболее мощных механизмов для уменьшения объема передаваемых по сети данных, притом внедряется этот механизм очень просто. Когда HTTP-сервер отправляет объект браузеру, он может дополнительно с ответом отправить специальный заголовок с меткой времени. Браузеры обычно хранят ресурс вместе с информацией об истечении его срока действия в локальном кэше. При последующих запросах к тому же объекту браузер сравнивает текущее время и метку времени у находящегося в кэше ресурса. Если метка времени указывает на дату в будущем, браузер загружает ресурс из кэша, не запрашивая его с сервера.

У всех браузеров существует ограничение на количество соединений на один хост, находящееся в интервале от 2 до 8 соединений на хост. Для увеличения скорости загрузки внешних объектов можно применять распределенную систему хранения и доставки контента CDN. CDN – это множество веб-серверов, разнесенных географически для достижения максимальной скорости отдачи контента клиенту. Сервер, который непосредственно будет отдавать контент пользователю, выбирается на основании некоторых показателей. Например, выбирается сервер с временем отклика. Стоит отметить, что браузер кэширует даже js-файлы, и если пользователи посещали сайты на которых используется такой метод, то эта библиотека уже есть в кэше браузера, и он не будет загружать ее снова. Одним из таких CDN является Google Libraries.

Многие известные ресурсы делают оптимизацию изображений. Оптимизировать изображение можно двумя способами: используя программы или специальные онлайн сервисы для сжатия. В первом случае потребуются определенные знания для работы с той или иной программой, а вот воспользоваться онлайн сервисами может каждый. Также не нужно изменять размер изображения при помощи стилей. Это тоже негативно влияет на скорость загрузки страницы. Если имеется изображение размером 500x500px, а вставить на сайт нужно изображение с размером 100x100px, то лучше изменить размер оригинальной картинки при помощи графического редактора. Чем меньший вес картинки, тем меньше времени потребуется для ее загрузки.

И последним полезным способом увеличения загрузки будет использование gzip-сжатия. Как показали проведенные исследования, gzip-сжатие текстового файла «на лету» в 95–98% случаев позволяет сократить время на передачу файла браузеру. Если хранить архивированные копии файлов на сервере, то соединение в общем случае удается освободить в 3-4 раза быстрее. Начиная с версии протокола

HTTP/1.1, веб-клиенты указывают, какие типы сжатия они поддерживают, устанавливая заголовок Accept-Encoding в HTTP-запросе. Если веб-сервер видит такой заголовок в запросе, он может применить сжатие ответа одним из методов, перечисленных клиентом. При выдаче ответа посредством заголовка Content-Encoding сервер уведомляет клиента о том, каким методом сжимался ответ. Переданные таким образом данные меньше первоначальных примерно в 5 раз, и это существенно ускоряет их доставку. Однако здесь есть один недостаток: увеличивается нагрузка на веб-сервер. Но вопрос с сервером всегда можно решить[2].

В последние годы мир веб-разработки стал существенно тяготеть к клиентским приложениям. Браузеры стали настолько быстрыми и приобрели такое феноменальное количество возможностей, что клиентская сторона стала не проще, а даже иногда намного сложнее, чем серверная составляющая. Именно на фоне увеличившегося внимания к клиентской составляющей веб-сайтов и зародилась ее оптимизация, оптимизация скорости загрузки, отображения и функционирования веб-сайтов в браузерах конечных пользователей. На данный момент направление это новое и весьма перспективное для изучения и прикладного использования.

ЛИТЕРАТУРА

1. Мациевский, Н.С. Реактивные веб-сайты. Клиентская оптимизация в алгоритмах и примерах : учеб. пособие / Н.С. Мациевский, Е.В. Степанищев, Г.И. Кондратенко. – М. : Интернет-Университет Информационных Технологий : БИНОМ. Лаборатория знаний, 2010. – 336 с. – Режим доступа: http://inethub.olvi.net.ua/ftp/pub/books/programming/web/html/reactivewebsites.v1.4.pdf. – Дата доступа: 29.09.2015.
2. Хабрахабр [Электронный ресурс] / Топ-10 советов о том, как увеличить скорость загрузки страницы. – Режим доступа: http://habrahabr.ru/post/137239/. – Дата доступа: 29.09.2015.

УДК 004.75

СПОСОБЫ ПОСТРОЕНИЯ РАСПРЕДЕЛЕННЫХ ВЕБ-СИСТЕМ

Ю.В. ЛАПТЕВ
(Представлено: канд. техн. наук, доц. Р.П. БОГУШ)

Рассматривается некоторые ключевые вопросы, которые следует учитывать при проектировании больших веб-сайтов, а также некоторые базовые компоненты, используемые для достижения этих целей. Основное внимание в статье уделяется анализу веб-систем.

Открытое программное обеспечение стало основным элементом при создании некоторых крупнейших веб-сайтов в последнее время. Создание и управление масштабируемого веб-сайта на примитивном уровне является простым соединением пользователей с удаленными ресурсами через Интернет. А доступ к ресурсам, которые рассредоточены на множестве серверов и являются звеном, обеспечивающим масштабируемость веб-сайта. Время, потраченное на планирование построения веб-службы на начальном этапе может помочь в дальнейшем. На проектирование крупномасштабных веб-систем могут влиять следующие принципы: производительность, стоимость, надежность, доступность, масштабируемость и управляемость.

Скорость работы веб-сайта влияет на удовлетворенность пользователей, а также ранжирование поисковыми системами. В результате нужно стремится к созданию системы, которая оптимизирована для быстрых ответов. Система должна быть надежной. Это означает, что определенный запрос на получение данных должен возвращал эти данные, а в случае изменения данных должен возвращать новые данные. Пользователи должны быть уверенным, что данные не будут потеряны. Разработка доступных к отказу систем является фундаментальным технологическим требованием. Высокая доступность в распределенных системах требует внимательного рассмотрения избыточности для ключевых компонентов, а также быстрого восстановления после системных отказов. Для крупных онлайн-магазинов недоступность даже в течении небольшого времени может привести к огромным потерям дохода. Очень важными являются усилия, направленные на увеличение пропускной способности для обработки больших объемов информации, что и является масштабируемостью. Также к масштабируемости относятся наращение емкости запоминающих устройств и увеличение количества транзакций, обрабатываемых в единицу времени. Проектирование системы, которая проста в эксплуатации еще один важный фактор. Для обеспечения управляемости необходимо рассмотреть вопросы диагностики возникающих проблем, легкости проведения обновлений или модификаций [1].