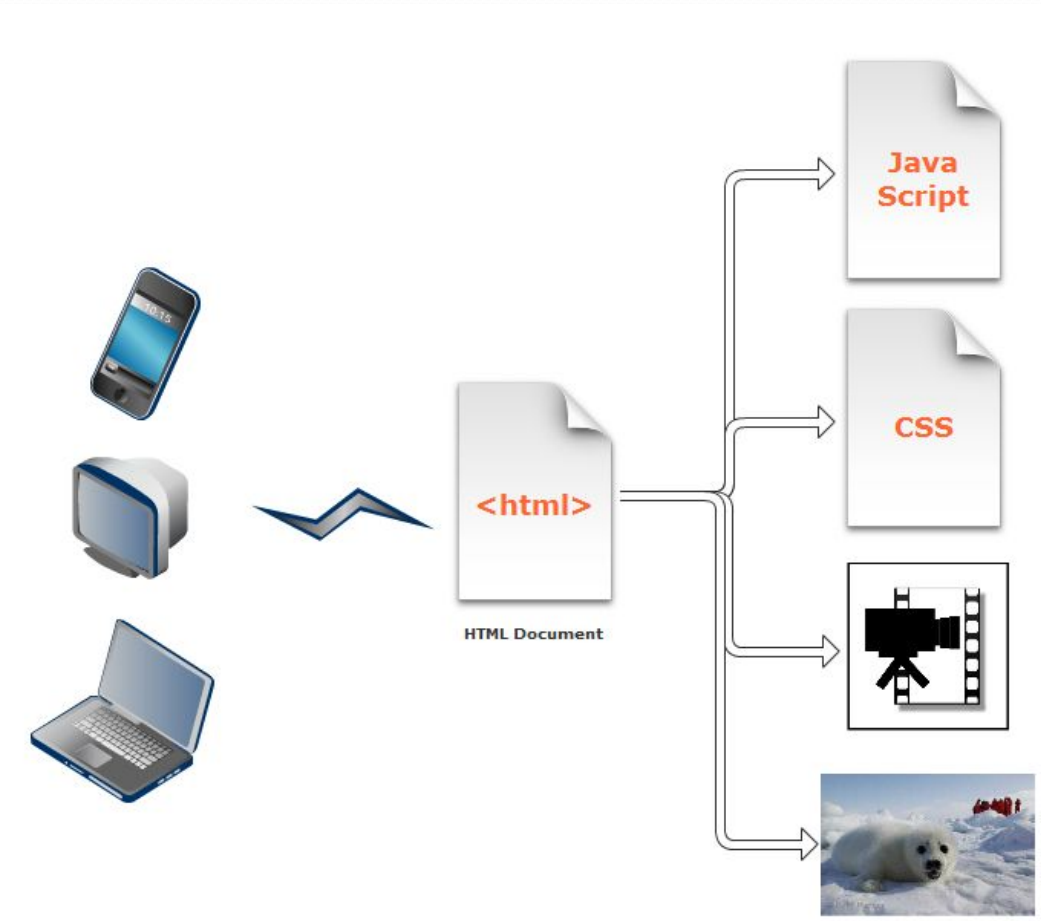
Андрей Кулешов

Деловые решения

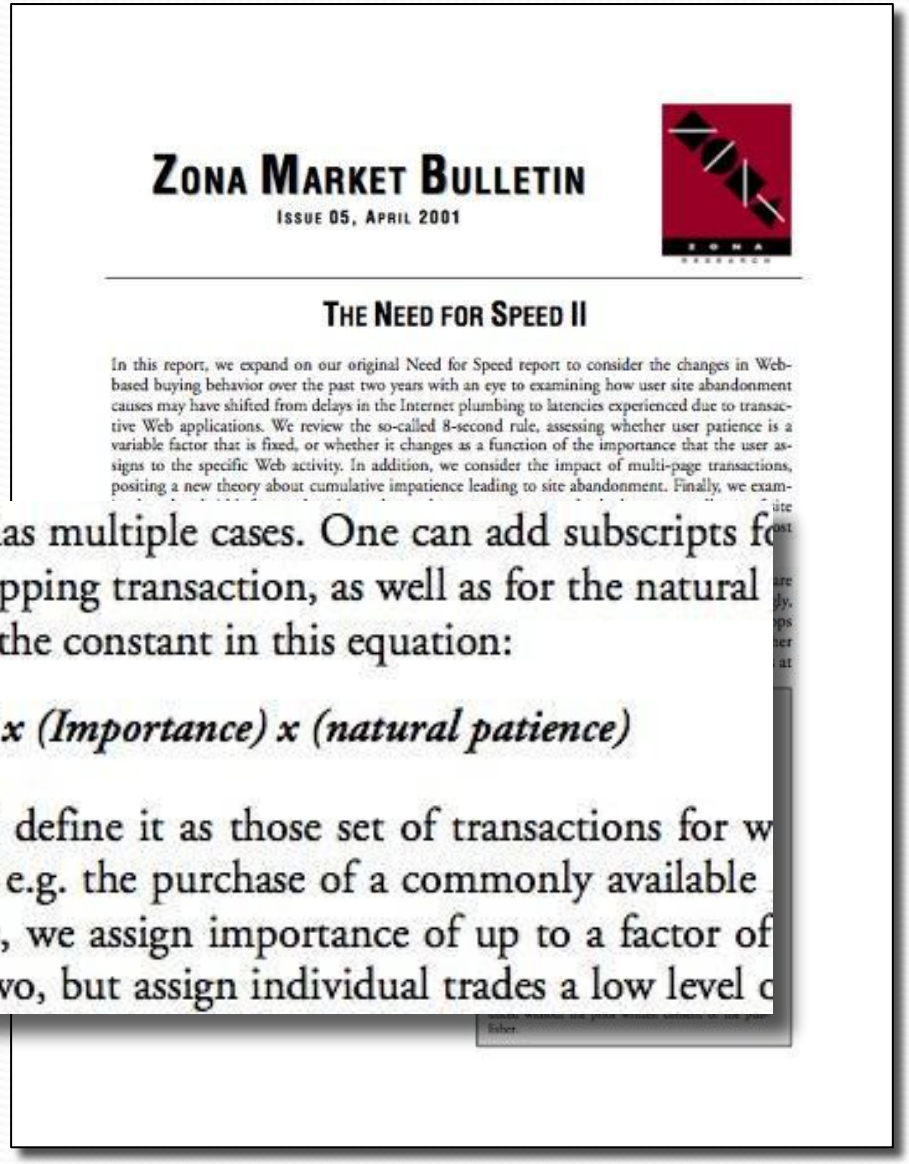
Про что мы говорим?



- Страница
- И множество связанных с ней ресурсов

Зачем городить огород?

Насколько терпеливы пользователи?



ZONA MARKET BULLETIN

ISSUE 05, APRIL 2001

THE NEED FOR SPEED II

In this report, we expand on our original Need for Speed report to consider the changes in Web-based buying behavior over the past two years with an eye to examining how user site abandonment causes may have shifted from delays in the Internet plumbing to latencies experienced due to transactive Web applications. We review the so-called 8-second rule, assessing whether user patience is a variable factor that is fixed, or whether it changes as a function of the importance that the user assigns to the specific Web activity. In addition, we consider the impact of multi-page transactions, positing a new theory about cumulative impatience leading to site abandonment. Finally, we exam-

Patience, as we indicated earlier, also has multiple cases. One can add subscripts fo assigned importance of the type of shopping transaction, as well as for the natural the user. Our familiar 8-second rule is the constant in this equation:

Patience = (8 seconds) x (Tenacity) x (Importance) x (natural patience)

We set normal importance at 1.0 and define it as those set of transactions for w most inclined to hop to a competitor, e.g. the purchase of a commonly available the items are harder-to-find specialties, we assign importance of up to a factor of stock trades an importance factor of two, but assign individual trades a low level c

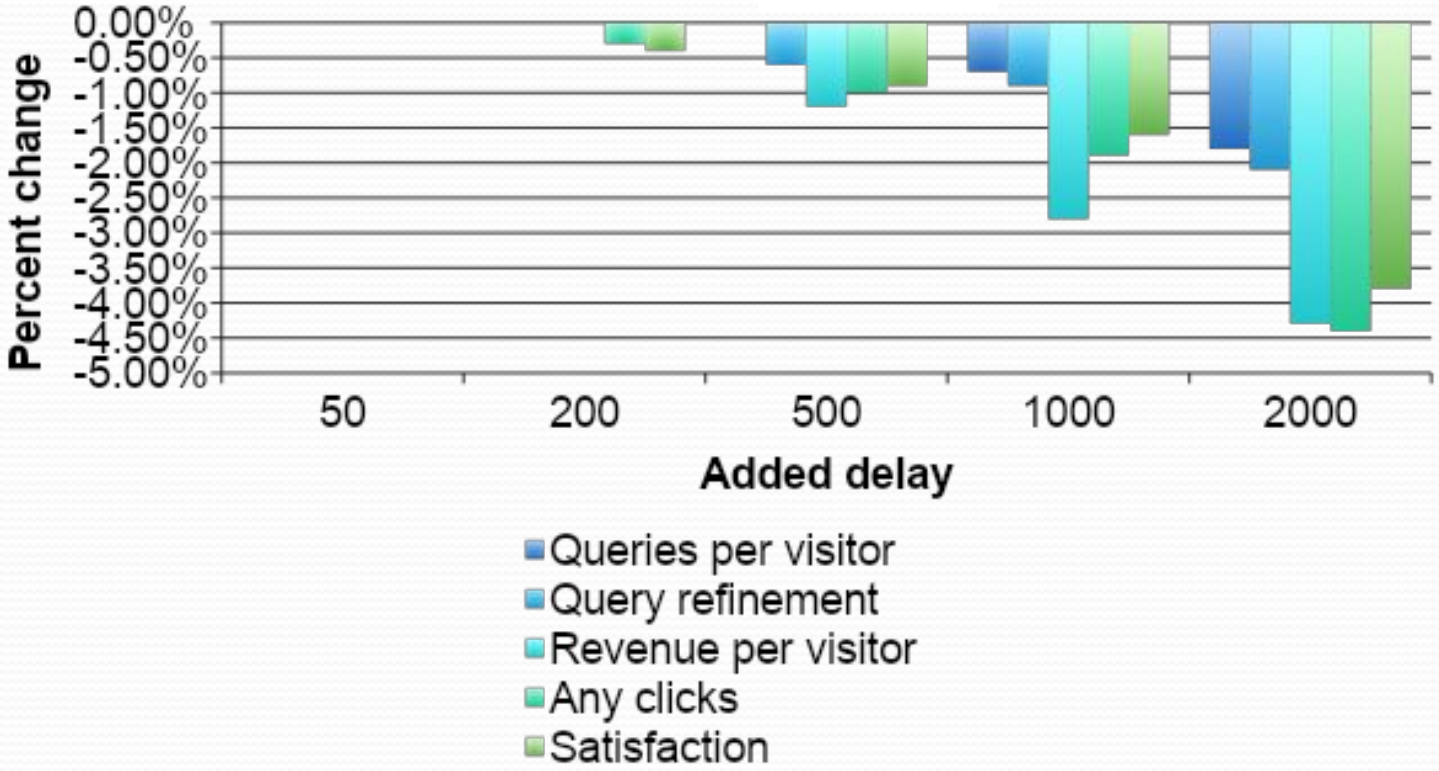
* Why Web Performance Matters (Richard Campbell)

Три числа терпения

- 0.1 секунды
  После этой границы человек начинает замечать задержку
- 1 секунда
  На этой границе человек начинает подозревать, что с системой что-то не так. Теряется чувство прямого взаимодействия с компьютером
- 10 секунд
  «Время смерти» – человек начинает подозревать, что система не работает. Перезагружает страницу второй раз, переключается на другие задачи, идёт делать чай

http://www.useit.com/papers/responsetime.html

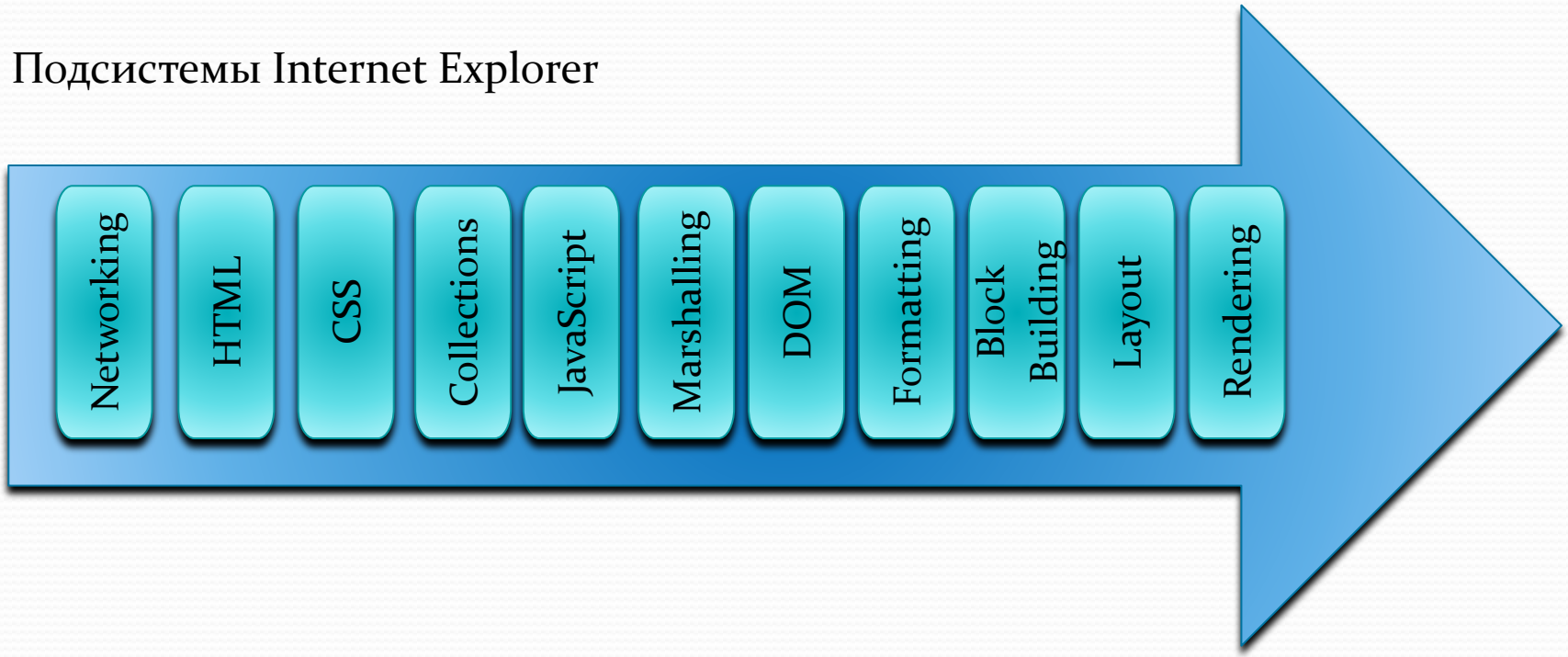
Изменение показателей при увеличении времени ожидания



* Why Web Performance Matters (Richard Campbell)

На что тратится время?

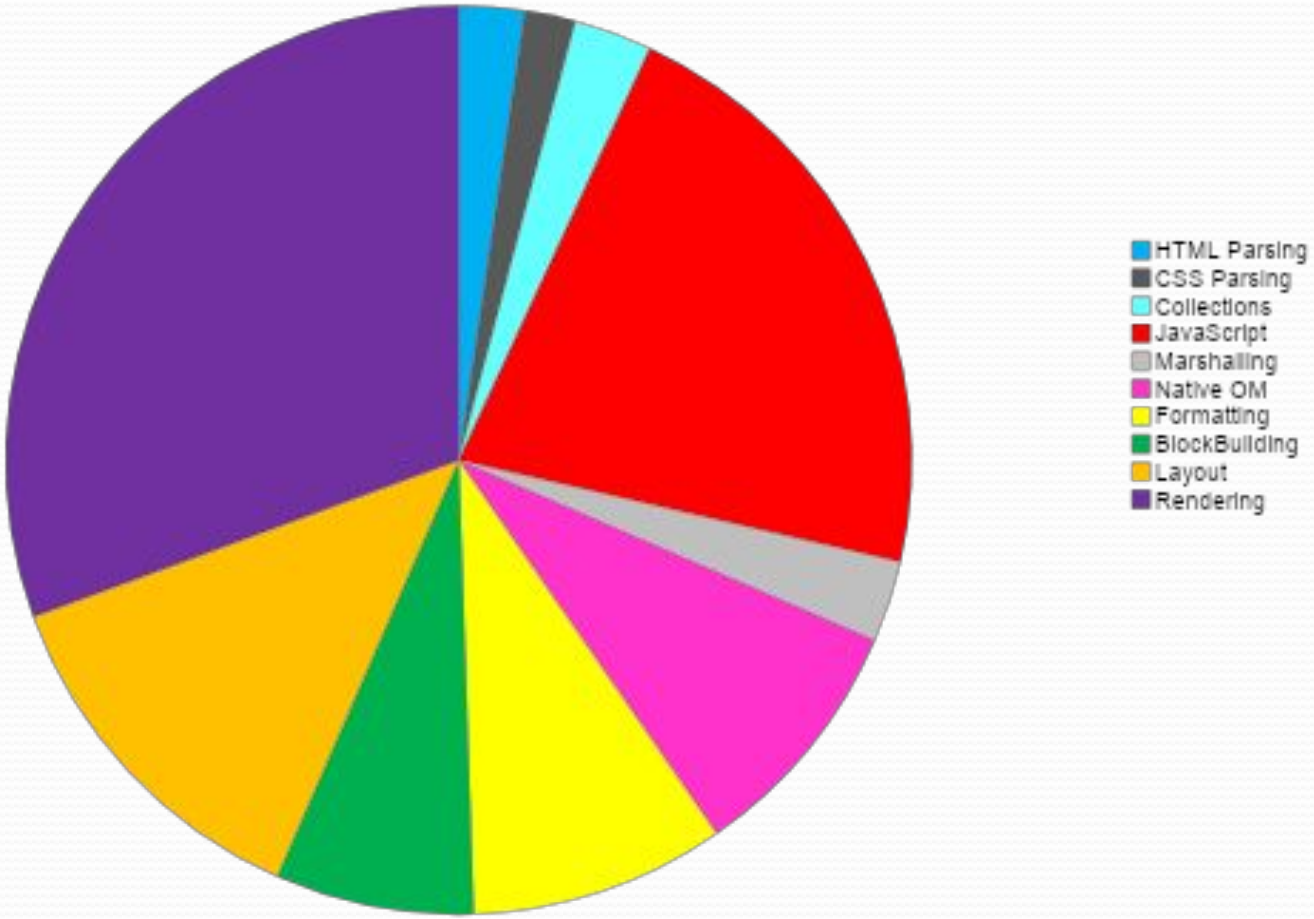
Подсистемы Internet Explorer



50 Performance Tricks to Make Your HTML5 Web Sites Faster (Jason Weber)

На что тратится время?

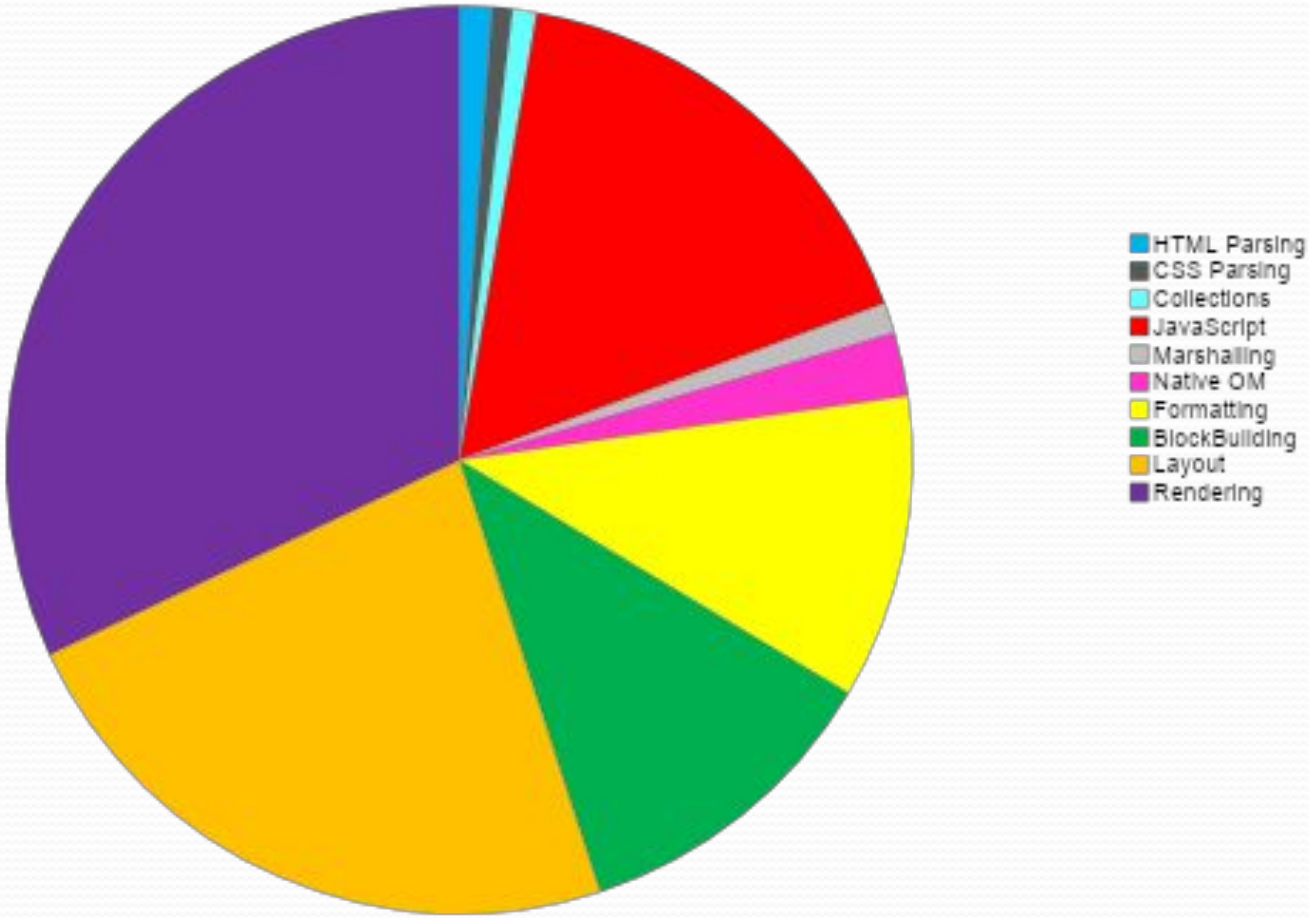
Средние значения для пяти новостных сайтов



50 Performance Tricks to Make Your HTML5 Web Sites Faster (Jason Weber)

На что тратится время?

Средние значения для нескольких AJAX-сайтов



50 Performance Tricks to Make Your HTML5 Web Sites Faster (Jason Weber)

Кто же побеждает?

- Побеждают не лучшие в чем-то одном
- Побеждают те, кто последовательно хороши во всём:
  - объём информации для скачивания;
  - количество DOM-элементов;
  - количество CSS-правил
  - количество изображений;
  - количество блоков JavaScript-кода;
  - количество строк JavaScript-кода;
  - и те, кто всё это правильно использует.

Начинается всё с Сети…

- Несмотря на рост пропускной способности каналов, она все равно остаётся ограниченной
- Особенно если учесть стремительно растущий рынок мобильных устройств
- Поэтому два основных принципа:
  - уменьшаем количество ресурсов
  - уменьшаем размер ресурсов

Чем меньше нужно скачать – тем лучше

- Минимизируйте количество используемых ресурсов

Не следует множить сущее без необходимости

Уильям Оккам

Избегайте перенаправлений

Request

```
GET / HTTP/1.1
Host: getdev.net
```

Response

```
HTTP/1.1 303 See Other
Location:
http://www.getdev.net/
```

Демо

Request? Response? Headers?

Сжатие на сервере

```
Response details
HTTP/1.1 200 OK
Content-Encoding:
gzip
Server:
Microsoft-IIS/7.5
```

- Все современные браузеры поддерживают приём сжатого содержимого (gzip, deflate)
- Резко уменьшается объем передаваемых по сети данных
- Поддерживается всеми основными серверами (IIS, Apache, nginx)
- Единственный минус – процессорное время. Но он с лихвой перевешивается выигрышем в траффике.

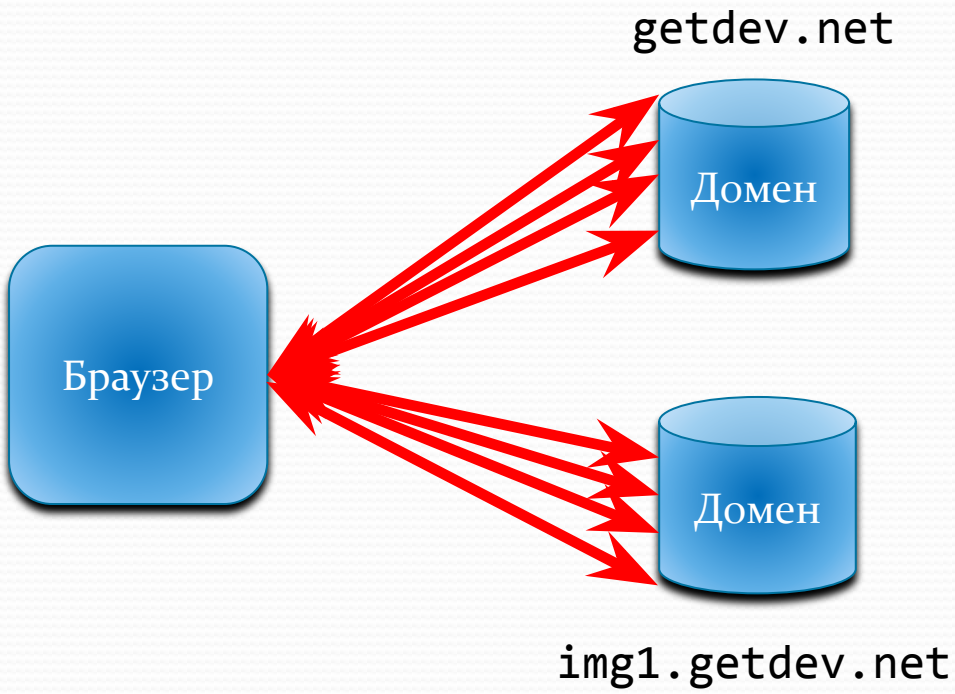
Кэширование

```
Response details
HTTP/1.1 200 OK
Content-Type:
image/jpeg
Expires: Sat, 31 Oct
2020 00:00:00 GMT
Last-Modified: Mon, 10
Oct 2011 18:55:14 GMT
ETag: "1fc57257e871:0“
Response details
HTTP/1.1 304 Not
Modified
Last-Modified: Mon, 10
Oct 2011 18:55:14 GMT
```

- Все ваши ресурсы с меняющимся содержим не должны быть закэшированы
- Все ваши ресурсы с постоянным содержим должны быть закэшированы навсегда
- ETag и Last-Modified
- Регистр в названиях ресурсов icon.png и Icon.png – один файл, но два скачивания…

Демо
Кэширование и сжатие на IIS

Вынесение ресурсов на несколько доменов



- Большинство современных браузеров загружают с одного домена одновременно 6-10 ресурсов
- Распределив ресурсы по нескольким доменам – получим больше одновременных загрузок
- Маленькая прибавка к производительности – на эти домены не будут пересылаться cookies при каждом запросу

Использование Content Delivery Network (CDN)

- Может быть использовано для статического контента (логотипы, картинки), и при некотором усердии – для динамического
- Пользователь скачивает ресурс с сервера, наиболее близкого к нему
- Если это популярная JS-библиотека – есть шанс, что у пользователя она уже скачана, и её не придётся качать заново

Использование спрайтов

- ImageSprite – одна большая картинка, которая содержит в себе множество маленьких
- И множество CSS-правил, которые хранят координаты и размер каждой из них
- Общий размер для скачивания уменьшается
- Нужно скачать только один файл

http://www.google.ru/intl/ru/options/
http://www.google.ru/options/i12.png

http://css-tricks.com/158-css-sprites/

Демо
Использование спрайтов
Использование нескольких доменов

Скачали данные – рендерим страницу

- Что тоже нужно делать быстро
- И иногда достаточно казаться, а не быть

CSS-файлы – в начале страницы

- Сразу же по получении CSS файла браузер начинает рендерить красивую картинку
- Пользователь быстрее получает сайт, приближенный по виду и разметке к финальному состоянию
- Загрузка других ресурсов при этом не блокируется

Не делайте вложенных и встроенных CSS-стилей

```
<html>
<head>
  <title>Test</title>
</head>
<body>
  <style>
    .item { color:#009900;}
  </style>
  <div class='item'>
    MyItem
  </div>
  <div style='color:Red;'>
    MyItem 2
  </div>
 </body>
</html>
```

- Это очень просто и быстро – сделать стиль для одного-единственного элемента встроенным в страницу
- Но на этапе рендеринга это ухудшает производительность

Используйте PNG, JPEG и JPEG-XR

- JPEG – для фотографий
- PNG – для всего остального
- GIF – устаревший формат – занимает больше места, потребляет больше CPU
- JPEG-XR – для фотографий высокого разрешения. Действительно высокого ☺

… и показывайте их в реальном размере

```
<!-- photo 800x600 -->
<img width="80px"
height="60px"
src="wallpaper.jpg" />
```

- Скачиваем больше, чем нужно
- Тратим процессорное время на изменение размера
- А ведь достаточно всего один раз изменить размер

Используйте CSS 3 и HTML5

Поддерживается большинством современных браузеров

- Для градиентов

```
-ms-gradient(linear, 50% 50%, 0% 34%, from(#666666), to(#666666), color-stop(.3,#333333))
-webkit-gradient(linear, 50% 50%, 0% 34%, from(#666666), to(#666666), color-stop(.3,#333333))
-moz-gradient(linear, 50% 50%, 0% 34%, from(#666666), to(#666666), color-stop(.3,#333333))
```

- Для скругления углов

```
border-radius:18px;
-webkit-border-radius: 100px;
-moz-border-radius: 100px;
```

- Для рисования на канве

И теперь можно запускать скрипты

- Запускать их лучше всего в тот момент, когда DOM уже полностью скачан.
  Внешние ресурсы в это время, скорее всего, ещё грузятся, но ждать их не имеет смысла

JS-файлы – в конце страницы

- Пока загружается и исполняется JavaScript файл – не производится ни скачивание, ни выполнение никаких других ресурсов
  (по стандарту. На практике, большинство браузеров скачивают ресурсы, но не выполняют их)
- Поэтому сначала лучше дать загрузиться всему остальному, чтобы пользователь увидел страницу, и лишь затем лезть со своим программированием
- Если очень-очень надо в начале – то помечаем аттрибутом defer (откладывающем выполнение)

Кэшируйте обращение к DOM

- $(“.class1”).show();
  $(“.class1”).hide();
  - два раза проходит по всему (возможно, очень большому!) дереву
  var x = $(“.class1”);
  x.show();
  x.hide();
  - почти в два раза эффективнее

Изменение DOM

- Если нужно внести изменения в страницу – сначала всё подготовьте, а потом уже один раз меняйте
  var content = getTitle() + getBody() + getFooter();
  myControl.innerHtml = content;

- innerHtml – самый быстрый способ сделать изменение

Минимизируйте JavaScript

```
/* this function is used to calculate sum of two numbers */
Function sumFunction (value1,  value2)
{
     var sum = value1 + value2;
     Return sum;
}
```

Минимизированный скрипт:

```
function sum(a, b){return a + b;}
```

- можно делать вручную (удачи)
- можно делать в момент выкладывания приложения
- можно делать во время исполнения приложения

Используйте Web Workers

```
var worker = new
Worker("worker_script.js");
worker.postMessage("Hello World!");
```

- Могут быть использованы для долговременных расчетов или для фоновых операций
- Не замедляют пользовательский интерфейс

Демо Минифицированный ЯваСкри

- Cassette for ASP.NET MVC by Andrew Davey
  https://github.com/andrewdavey

А можно как-то вот это всё автоматизировать?

- Есть инструменты.
- Встроенные средства разработчика во всех браузерах
- Плагины от Гугла и Яху
  http://veerasundar.com/blog/2009/06/google-page-speed-firefox-plugin-for-improving-website-performance/
  YSlow - https://addons.mozilla.org/ru/firefox/addon/yslow/
- Инструменты минимизации
  AjaxMin - http://ajaxmin.codeplex.com/
  YUI Compressor - http://developer.yahoo.com/yui/compressor/
  JSMin
  … и многие другие

Вопросы?
Внимательно слушаю! ☺

Андрей Кулешов
«Деловые решения»
Директор

akuleshov@solforbiz.com
http://www.solforbiz.com

akuleshov.tula

Специально для http://GetDev.NET

Интересное чтение

- Best Practices for Speeding Up Your Web Site – Yahoo
  http://developer.yahoo.com/performance/rules.html
- Steve Souders – эксперт из Google
  http://www.stevesouders.com/
- Top 10 Client-Side Performance Problems in Web 2.0
  http://blog.dynatrace.com/2010/08/25/top-10-client-side-performance-problems-in-web-2-0/

Интересное видео

- Why Web Performance Matters - Richard Campbell http://channel9.msdn.com/Events/TechEd/NorthAmerica/2011/DEV344
- 50 Performance Tricks to Make Your HTML5 Web Sites Faster - Jason Weber (Principal Program Manager Lead for Internet Explorer) http://channel9.msdn.com/events/MIX/MIX11/HTM01
- Повышение производительности клиентской части сайта с высокой нагрузкой – Евгений Чигиринский http://www.techdays.ru/videos/3708.html