Performance Grade: C (76)

1. Make fewer HTTP requests
   This page has 9 external JavaScript files.

2. Use a CDN

3. Add an Expires header
   These components do not have a far future Expires
1. Speed - First and foremost, we believe that speed is more than a feature. Speed is the most important feature.
Importance of performance

- Yahoo!: 400 ms slower = 5-9% drop in full-page traffic
## Server Delays Experiment: Results

<table>
<thead>
<tr>
<th>Time Delay (ms)</th>
<th>Distinct Queries/User</th>
<th>Query Refinement</th>
<th>Revenue/User</th>
<th>Any Clicks</th>
<th>Satisfaction</th>
<th>Time to Click (increase in ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50ms</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>200ms</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-0.3%</td>
<td>-0.4%</td>
<td>500</td>
</tr>
<tr>
<td>500ms</td>
<td>-90.6%</td>
<td>-1.2%</td>
<td>-1.0%</td>
<td>-0.9%</td>
<td>1200</td>
<td></td>
</tr>
<tr>
<td>1000ms</td>
<td>-0.7%</td>
<td>-0.9%</td>
<td>-2.8%</td>
<td>-1.9%</td>
<td>-1.6%</td>
<td>1900</td>
</tr>
<tr>
<td>2000ms</td>
<td>-1.8%</td>
<td>-2.1%</td>
<td>-4.3%</td>
<td>-4.4%</td>
<td>-3.8%</td>
<td>3100</td>
</tr>
</tbody>
</table>

* - Means no statistically significant change

- **Strong negative impacts**
- **Roughly linear changes with increasing delay**
- **Time to Click changed by roughly double the delay**
Search Traffic Impact

<table>
<thead>
<tr>
<th>Type of Delay</th>
<th>Delay (ms)</th>
<th>Experiment Duration (weeks)</th>
<th>Impact on Average Daily Searches Per User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-header</td>
<td>50</td>
<td>4</td>
<td>Not measurable</td>
</tr>
<tr>
<td>Pre-header</td>
<td>100</td>
<td>4</td>
<td>-0.20%</td>
</tr>
<tr>
<td>Post-header</td>
<td>200</td>
<td>6</td>
<td>-0.29%</td>
</tr>
<tr>
<td><strong>Post-header</strong></td>
<td><strong>400</strong></td>
<td><strong>6</strong></td>
<td><strong>-0.59%</strong></td>
</tr>
<tr>
<td>Post-ads</td>
<td>200</td>
<td>4</td>
<td>-0.30%</td>
</tr>
</tbody>
</table>

- Increase in abandonment heuristic = less satisfaction
  - Abandonment heuristic measures if a user stops interacting with search engine before they find what they are looking for
- Active users (users that search more often a priori) are more sensitive
...shaved 2.2 seconds off the average page load time and increased download conversions by 15.4%!
Performance Summary

- Conversion Rate: +7% - 12%
- Page View’s: +25%
- US SEM Sessions: +8%
- Bizrate.co.uk SEM Sessions: +120%
- Infrastructure Required (US): -50% (200 vs 402 nodes)
- Availability: 99.71% → 99.94%
- Product Velocity: +225%
- Release Cost: $1,000’s → $80
Improving Performance: Gzip

- Payload reduced in some cases 15x (typically in half)

<table>
<thead>
<tr>
<th>Empty Cache</th>
<th>Primed Cache</th>
<th>Empty Cache</th>
<th>Primed Cache</th>
</tr>
</thead>
<tbody>
<tr>
<td>804.8K 1HTML/Text</td>
<td>804.8K 1HTML/Text</td>
<td>51.6K 1HTML/Text</td>
<td>51.6K 1HTML/Text</td>
</tr>
<tr>
<td>284.5K 3JavaScript Files</td>
<td>284.5K 3JavaScript Files</td>
<td>0.0K 1XMLHttpRequest</td>
<td>0.0K 1XMLHttpRequest</td>
</tr>
<tr>
<td>73.4K 2Stylesheet Files</td>
<td>73.4K 2Stylesheet Files</td>
<td>63.9K 3JavaScript Files</td>
<td>0.0K 1XMLHttpRequest</td>
</tr>
<tr>
<td>8.0K 2CSS Images</td>
<td>0.0K 2CSS Images</td>
<td>15.4K 2Stylesheet Files</td>
<td>0.0K 2CSS Images</td>
</tr>
<tr>
<td>55.0K 60Images</td>
<td>0.0K 60Images</td>
<td>21.9K 23CSS Images</td>
<td>0.0K 60Images</td>
</tr>
<tr>
<td>1229.8K Total size</td>
<td>1102.8K Total size</td>
<td>56.3K 61Images</td>
<td>56.3K 61Images</td>
</tr>
<tr>
<td>86 HTTP requests</td>
<td>86 HTTP requests</td>
<td>209.5K Total size</td>
<td>209.5K Total size</td>
</tr>
<tr>
<td></td>
<td></td>
<td>91 HTTP requests</td>
<td>91 HTTP requests</td>
</tr>
</tbody>
</table>

- User experience performance improvement: 13% - 25%

- Network outbound traffic cut in half
...we've decided to take site speed into account in our search rankings.
WPO  Web Performance Optimization

drives traffic
improves UX
increases revenue
reduces costs
top 10 WPO predictions

#10: fast-by-default

- CMSs
- PHP, Rails, Python
- cloud - AWS, GAE
- JS libraries
- browsers, servers, proxies
top 10 WPO predictions

#9: visibility into the browser

- JS, CSS, paint, DOM
- dynaTrace
- Speed Tracer
top 10 WPO predictions

#8: consolidation

- tools
- metrics
- services
top 10 WPO predictions

#7: TCP, HTTP

- SPDY
- pipelining
- resource packages
top 10 WPO predictions

#6: standards
  • terminology
  • timing
  • benchmarks
  • testing
top 10 WPO predictions

#5: industry organizations
- professional groups
- training, certification
- standards bodies
- cooperatives
top 10 WPO predictions

#4: data
- Internet Performance Archive
- network parameters
- DNS
- cache
top 10 WPO predictions

#3: green
  • Shopzilla - half the servers
  • Netflix - half the bandwidth
  • power consumption
  • carbon footprint
top 10 WPO predictions

#2: mobile
  • metrics
  • long poles
  • root causes
  • solutions, best practices
  • tools, automation
  • evangelism, case studies
top 10 WPO predictions

#1: speed’s a differentiator
  • devices
  • vendor selection
  • reviews
  • user loyalty
top 10 WPO predictions

10. fast-by-default
9. visibility into the browser
8. consolidation
7. TCP, HTTP
6. standards
5. industry organizations
4. data
3. green
2. mobile
1. speed’s a differentiator
P3PC
Performance of 3rd Party Content
Ads
Widgets
Analytics
Performance of 3rd Party Content

Ads, widgets, and analytics are a major cause for slow web sites. P3PC is a project focused on analyzing the performance of 3rd party content. The goal is to find the key wins to evangelize to make 3rd party content faster.

The table below summarizes my findings. Click on the links to read the detailed analysis of each snippet. As I do new analyses I'll add them to this table. You can leave suggestions for other 3rd party content to investigate on my original P3PC blog post.

You can help out this project by clicking on the Compare button and measuring how long these snippets take to load in your browser.

<table>
<thead>
<tr>
<th>snippet blog post</th>
<th>impact on page</th>
<th>Page Speed</th>
<th>YSlow</th>
<th>doc. write</th>
<th>total reqs</th>
<th>total xfer size</th>
<th>JS ungzip</th>
<th>DOM elems</th>
<th>median Δ load time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digg widget</td>
<td>big</td>
<td>90</td>
<td>84</td>
<td>y</td>
<td>9</td>
<td>52 kB</td>
<td>107 kB</td>
<td>84</td>
<td>669 ms</td>
</tr>
<tr>
<td>Facebook Sharer</td>
<td>small</td>
<td>90</td>
<td>92</td>
<td>n</td>
<td>5</td>
<td>8 kB</td>
<td>7 kB</td>
<td>15</td>
<td>145 ms</td>
</tr>
<tr>
<td>Google Analytics</td>
<td>small</td>
<td>91</td>
<td>99</td>
<td>n</td>
<td>2</td>
<td>18 kB</td>
<td>24 kB</td>
<td>2</td>
<td>27 ms</td>
</tr>
<tr>
<td>BuySellAds*</td>
<td>small</td>
<td>81</td>
<td>92</td>
<td>n</td>
<td>3</td>
<td>7 kB</td>
<td>14 kB</td>
<td>9</td>
<td>44 ms</td>
</tr>
<tr>
<td>Google AdSense*</td>
<td>big</td>
<td>87</td>
<td>84</td>
<td>y</td>
<td>8</td>
<td>41 kB</td>
<td>76 kB</td>
<td>9</td>
<td>96 ms</td>
</tr>
<tr>
<td>ValueClick*</td>
<td>med</td>
<td>89</td>
<td>98</td>
<td>y</td>
<td>3</td>
<td>2 kB</td>
<td>1 kB</td>
<td>1</td>
<td>na**</td>
</tr>
<tr>
<td>Quantcast</td>
<td>small</td>
<td>93</td>
<td>98</td>
<td>n</td>
<td>2</td>
<td>3 kB</td>
<td>3 kB</td>
<td>2</td>
<td>118 ms</td>
</tr>
<tr>
<td>Collective Media*</td>
<td>big</td>
<td>86</td>
<td>90</td>
<td>y</td>
<td>6</td>
<td>8 kB</td>
<td>9 kB</td>
<td>7</td>
<td>na**</td>
</tr>
<tr>
<td>Glam Media*</td>
<td>big</td>
<td>89</td>
<td>83</td>
<td>y</td>
<td>11</td>
<td>68 kB</td>
<td>63 kB</td>
<td>7</td>
<td>na**</td>
</tr>
</tbody>
</table>

* Stats for ads only include the ad framework and not any ad content.
** It's not possible to gather timing stats for snippets with live ads.

Leave a comment on the related P3PC blog post.
appendChild or insertBefore?

document.getElementsByTagName('head')[0].appendChild(domscript);

Souders (May 2008)

document.documentElement.firstChild.appendChild(ga);

- Google Analytics (Dec 2009)

s = document.getElementsByTagName('script')[0];
s.parentNode.insertBefore(ga, s);

- Google Analytics (Feb 2010)

flickr.com/photos/amodiovalerioverde/425333516/
appendChild or insertBefore?

```javascript
if (q.insertBefore) {
    var s = _get(q.insertBefore, id);
    if (s) {
        s.parentNode.insertBefore(n, s);
    }
} else {
    h.appendChild(n);
}
```

- YUI Loader 2.6.0 (2008)

flickr.com/photos/amodiovalerioverde/425333516/
appendChild or insertBefore?

b = this.getHardpoint();
f = document.createElement('script');
f.src = g;
f.type = 'text/javascript';
f.async = true;
b.appendChild(f);

getHardpoint: function() {
    var c, b =
    document.getElementsByTagName('head');
    if (b.length) { c = b[0]; }
    else c = document.body;
    this._hardpoint = c;
}

- Facebook (April 18, 2010)

flickr.com/photos/am odiovalериоverde/425333516/
appendChild or insertBefore?

head = document.getElementsByTagName("head")[0] || document.documentElement;

// Use insertBefore instead of appendChild to circumvent an IE6 bug.

// This arises when a base node is used (#2709 and #4378).

head.insertBefore(script, head.firstChild);

- jQuery

flickr.com/photos/amodiovalerioverde/425333516/
appendChild or insertBefore?

```javascript
var f = document.getElementsByTagName("script");
var b = f[f.length - 1]; // b is last script tag
if (b == null) { return; }
var i = document.createElement("script");
i.language = "javascript";
i.setAttribute("type", "text/javascript");
var j = "";
var g = document.createTextNode(j);
b.parentNode.insertBefore(i, b);
appendChild(i, j);

function appendChild(a, b) {
  if (null == a.canHaveChildren || a.canHaveChildren) {
    a.appendChild(document.createTextNode(b));
  } else {
    a.text = b;
  }
} - Collective Media
```

flickr.com/photos/amodiovalerioverde/425333516/
“Frag Tag” Alex Russell

<FRAG>
<script src="snippet.js"></script>
</FRAG>

doesn’t block rendering async document.write JavaScript sandboxing just a twinkle in my eye
browser wishlist
stevesouders.com/blog/2010/02/15/browser-performance-wishlist/

Frag tag
SPDY
non-blocking scripts
SCRIPT attributes
resource packages
border-radius
cache redirects
link prefetch
Web Timing spec
remote JS debugging
web sockets

History
progressive XHR
anchor ping
stylesheet, inline js
inline script defer
@import
@font-face
stylesheets, iframes
paint events
missing schema

flickr.com/photos/eole/380316678/
browser disk cache

default size is too small
  • IE: 8-50M
  • Firefox: 50M
  • Chrome: < 80M
  • Opera: 20M

eviction algorithm improvements
  • content-type: script > image
  • past impact: 1000 ms > 100 ms
  • preferred sites

stevesouders.com/cache.php
Browserscope is a community-driven project for profiling web browsers. The goals are to foster innovation by tracking browser functionality and to be a resource for web developers.

Gathering test results from users "in the wild" is the most important and useful feature of Browserscope - and you can participate!
HTTP Archive Format (HAR)
HAR to Page Speed

Generate Page Speed scores from an HTTP Archive file.

HAR file: [Browse] [Upload]

examples: CNN, Google search, Yahoo search, Wikipedia.

Create your own HAR files using Firebug and NetExport.

77 Page Speed

- 0 Minimize DNS lookups
- 33 Combine external CSS
- 40 Leverage browser caching
- 63 Minify JavaScript
- 74 Minify CSS
- 81 Combine external JavaScript
- 85 Optimize the order of styles and scripts
- 88 Specify a cache validator
- 93 Minify HTML
- 97 Enable compression
- 100 Avoid bad requests
- 100 Minimize redirects
- 100 Minimize request size
- 100 Serve resources from a consistent URL
- 100 Specify a character set early

HAR Viewer by Honza
Velocity Web Performance and Operations Conference
Fast by Default

Web performance and operations is an emerging discipline which requires incredible breadth, focusing less on specific technologies and more on how the entire system works together. While people often specialize in particular components, great engineers and developers understand web performance and operations in relation to the whole. The best are able to fly to the 50,000 foot view and see the entire system in motion and then zoom in to microscopic levels and examine the tiny movements of an individual part.

View the full Velocity schedule >>

Now in its third year, Velocity—the Web Performance and Operations conference from O’Reilly Media—is the premier conference that:

- Provides your web ops and dev teams direct access to the training, technologies and skills that will have the most immediate impact on your bottom line
- Gains you the keys to the “fast by default” toolkit for automating your web system's performance

REGISTER NOW AND SAVE

CONFERECE DISCOUNTS

Want to send several members of your company to Velocity? Are you a government employee? Instructor or student? Let our discounts help you make the case to attend.

Twitter (Tag: #velocityconf)

- I’m speaking at #velocityconf 2010. Join me and regist with vel10sp to save 20%
  http://en.oreilly.com/velocity2010
  8 hours ago from GoldsmithEric

- RT #velocityconf... just a few hours until #velocityconf early reg closes. by midnight!
  http://oreil.ly/cgwto
  10 hours ago from thebitsource

- For all of you last minute decision makers - just a few hours until #velocityconf early reg closes. Reg by midnight! http://oreil.ly/cgwto
Using site speed in web search ranking

Friday, April 09, 2010 at 11:00 AM
Webmaster Level: All

You may have heard that here at Google we're obsessed with speed, in our products and on the web. As part of that effort, today we're including a new signal in our search ranking algorithms: site speed. Site speed reflects how quickly a website responds to web requests.

Speeding up websites is important — not just to site owners, but to all Internet users. Faster sites create happy users and we've seen in our internal studies that when a site responds slowly, visitors spend less time there. But faster sites don't just improve user experience; recent data shows that improving site speed also reduces operating costs. Like us, our users place a lot of value in speed — that's why we've decided to take site speed into account in our search rankings. We use a variety of sources to determine the speed of a site relative to other sites.

If you are a site owner, webmaster or a web author, here are some free tools that you can use to evaluate the speed of your site:

- **Page Speed**, an open source Firefox/Firebug add-on that evaluates the performance of web pages and gives suggestions for improvement.
- **YSlow**, a free tool from Yahoo! that suggests ways to improve website speed.
- **WebPageTest** shows a waterfall view of your pages' load performance plus an optimization checklist.
- In Webmaster Tools, Labs > Site Performance shows the speed of your website as
Site performance

This page shows you performance statistics of your site. You can use this information to improve the speed of your site and create a faster experience for your users. Learn more

Performance overview
On average, pages in your site take 1.4 seconds to load (updated on Apr 13, 2010). This is faster than 83% of sites. These estimates are of medium accuracy (between 100 and 1000 data points). The chart below shows how your site’s average page load time has changed over the last few months. For your reference, it also shows the 20th percentile value across all sites, separating slow and fast load times.

Example pages
These are some example pages from your site and the time that they take to load in a browser (in seconds).

<table>
<thead>
<tr>
<th>URL</th>
<th>Load time</th>
</tr>
</thead>
<tbody>
<tr>
<td>/hpws/imagemap-no.php</td>
<td>1.0</td>
</tr>
<tr>
<td>/hpws/expiration.php</td>
<td>0.8</td>
</tr>
<tr>
<td>/cuzillion/</td>
<td>4.2</td>
</tr>
</tbody>
</table>
Both
combine scripts
combine stylesheets
add an Expires header
gzip responses
put stylesheets at the top
put scripts at the bottom
avoid CSS expressions
make JS and CSS external
reduce DNS lookups
minify JS and CSS
avoid redirects
remove duplicate scripts
make Ajax cacheable
reduce cookie size
use cookie-free domains
don't scale images

YSlow
use CSS sprites
use a CDN
configure ETags
use GET for Ajax requests
reduce # of DOM elements
no 404s
avoid image filters
optimize favicon

Page Speed
derfer loading JS
remove unused CSS
use efficient CSS selectors
optimize images
optimize order of CSS & JS
shard domains
leverage proxy caching
STEP 1 - ENTER TEST URL:

Label: (i.e. AOL News)

STEP 2 - CHOOSE TEST LOCATION

- Dulles, VA USA Provided by AOL
- San Jose, CA USA Provided by strangeloop
- Gloucester, UK Provided by DAEMON
- Jiangsu, China Provided by strangeloop
- Wellington, New Zealand Provided by optimize

STEP 3 - CHOOSE A CONFIGURATION

Browser: Internet Explorer 7

Connection: FIOS

Bandwidth:
- Down: 1500 Kbps
- Up: 384 Kbps
- Latency: 50 ms

Pending Tests (Backlog):

0

STEP 4 - TEST OPTIONS

Basic Settings Advanced Settings Auth Script Block Video
Test completed - 04/15/10 16:32:24 from Dulles, VA - 1.5Mbps ADSL

<table>
<thead>
<tr>
<th>Load Time</th>
<th>First Byte</th>
<th>Start Render</th>
<th>Document Complete</th>
<th>Fully Loaded</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.185s</td>
<td>0.322s</td>
<td>1.655s</td>
<td>2.185s 15 56 KB 244.83 Kbps</td>
<td>2.185s 15 56 KB</td>
</tr>
</tbody>
</table>

**Key Optimizations**

- Enable keep-alive: A
- Compress Text: A
- Compress Images: B
- Cache static content: F
- Combine js and css files: A
- Use a CDN: F

[click for optimization details](#)

**Test Results**

<table>
<thead>
<tr>
<th>Waterfall</th>
<th>Optimization Checklist</th>
<th>Screen Shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>First View (2.185s)</td>
<td><img src="chart.png" alt="Waterfall Chart" /></td>
<td><img src="screen.png" alt="Screen Shot" /></td>
</tr>
</tbody>
</table>
takeaways

speed matters

a lot is coming in WPO

guard against $3^{rd}$ party content

stevesouders.com/cache.php