High Performance Web Sites

14 rules for faster-loading pages

Steve Souders
souders@yahoo-inc.com

Tenni Theurer
tenni@yahoo-inc.com

Introduction

Exceptional Performance

started in 2004

quantify and improve the performance of all Yahoo! products worldwide

center of expertise

build tools, analyze data

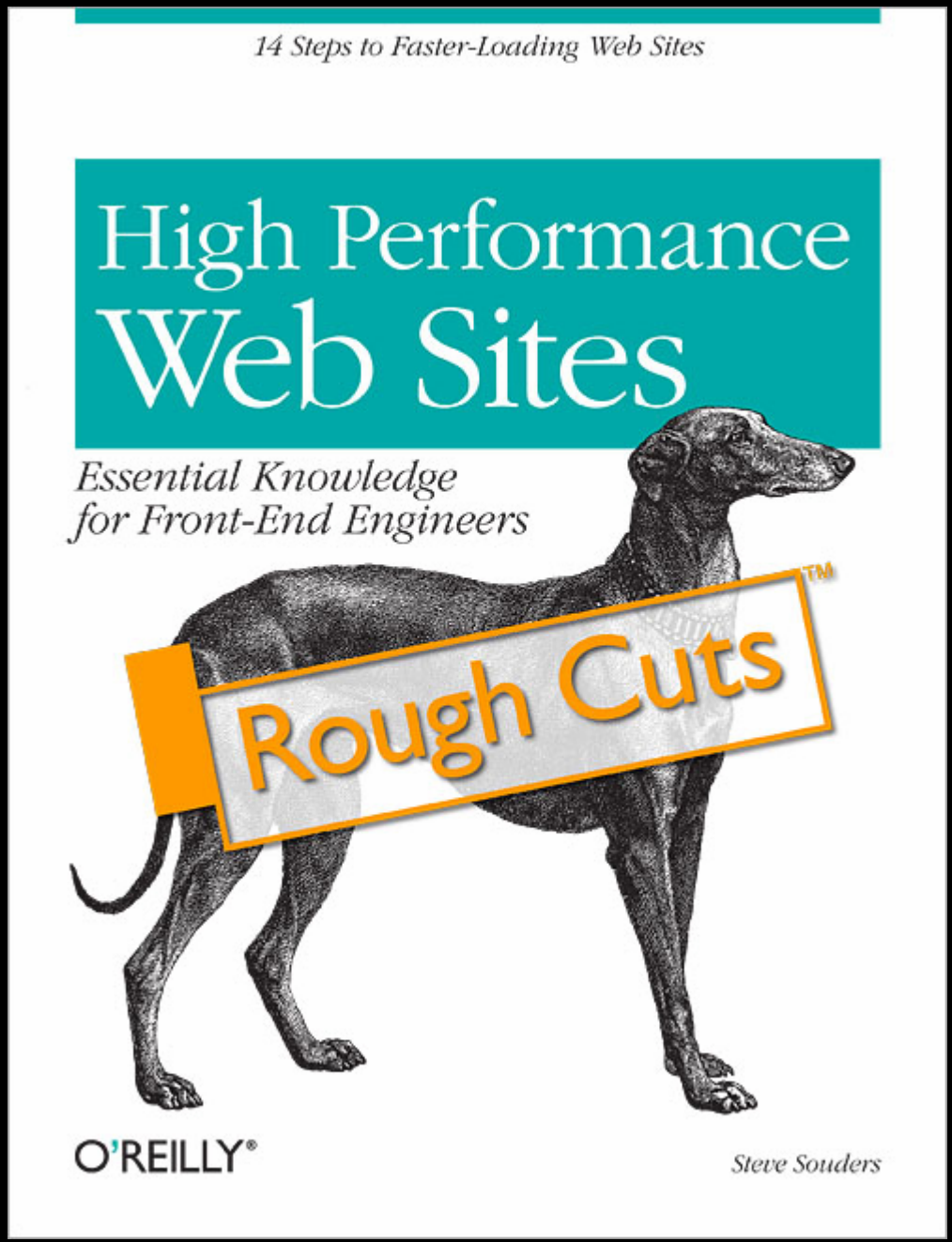
gather, research, and evangelize best practices

Scope

performance breaks into two categories
- response time
- efficiency

current focus is response time

of web products

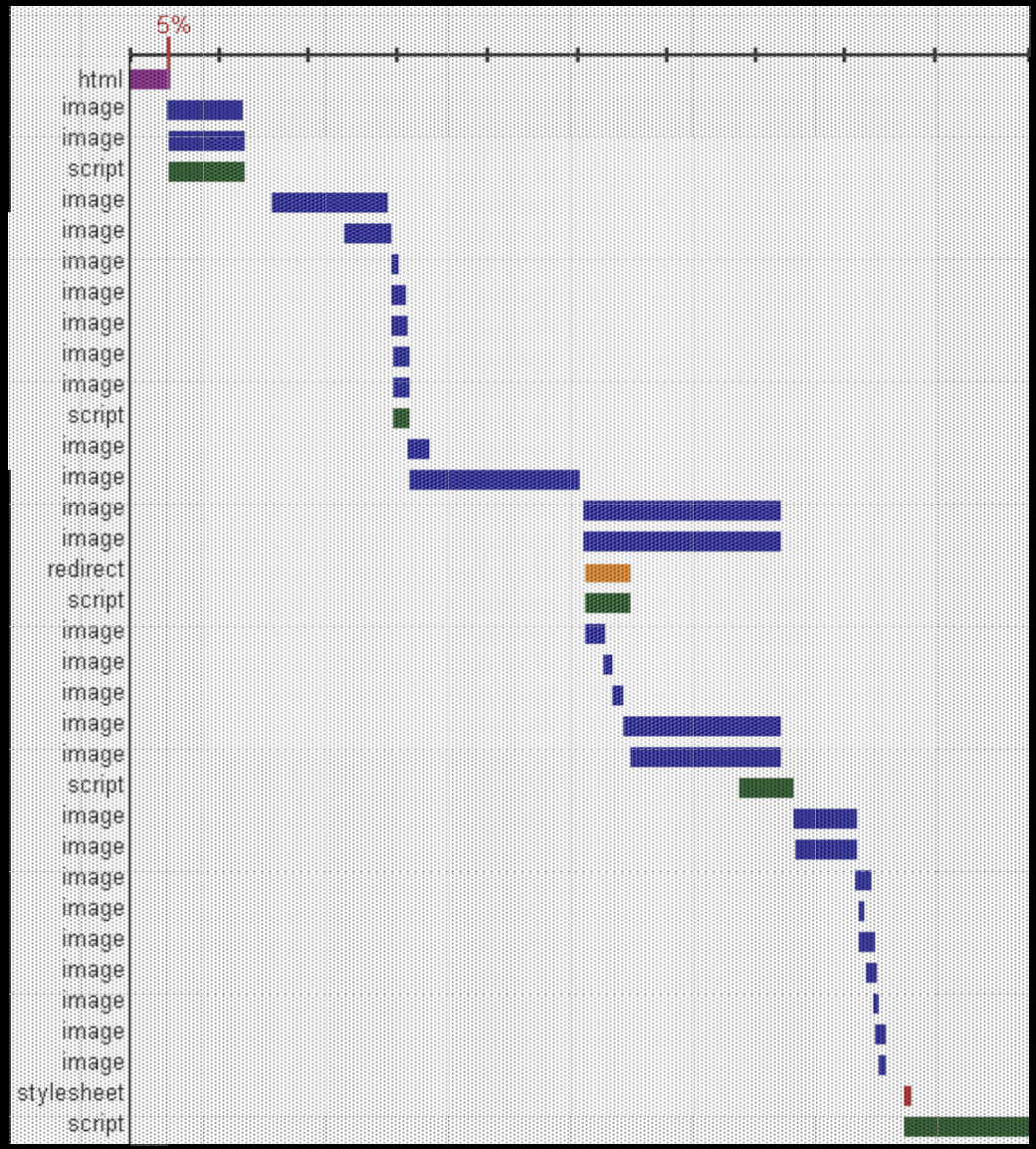


Rough Cuts: now

Hardcopy: Summer 2007

http://www.oreilly.com/catalog/9780596514211/

The Importance of Front-End Performance



Back-end vs. Front-end

	Empty Cache	Full Cache
amazon.com	82%	86%
aol.com	94%	86%
cnn.com	81%	92%
ebay.com	98%	92%
google.com	86%	64%
msn.com	97%	95%
myspace.com	96%	86%
wikipedia.org	80%	88%
yahoo.com	95%	88%
youtube.com	97%	95%

percentage of time spent on the front-end

The Performance Golden Rule

80-90% of the end-user response time is spent on the front-end. Start there.

- Greater potential for improvement
- Simpler
- Proven to work

Schedule

Performance Research

break

14 Rules

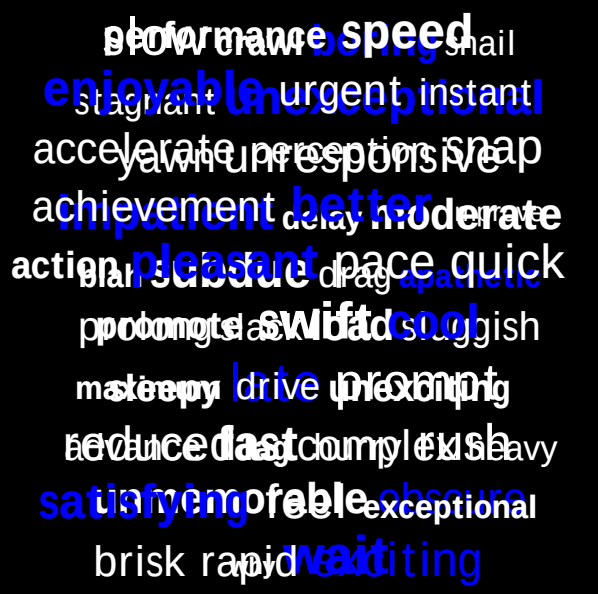
break

Case Studies

Live Analysis

Performance Research

perceived response time



what is the end user’s experience?

User Perception

Usability and perception are important for performance.

The user’s perception is more relevant than actual unload-to-onload response time.

Definition of "user onload" is undefined or varies from one web page to the next.

Performance Research, Part 1: What the 80/20 Rule Tells Us about Reducing HTTP Requests

November 28, 2006 at 12:56 pm by Tenni Theurer | In Development |

> *This is the first in a series of articles describing experiments conducted to learn more about optimizing web page performance. You may be wondering why you're reading a performance article on the YUI Blog. It turns out that most of web page performance is affected by front-end engineering, that is, the user interface design and development.*

It's no secret that users prefer faster web sites. I work in a dedicated team focused on quantifying and improving the performance of Yahoo! products worldwide. As part of our work, we conduct experiments related to web page performance. We are sharing our findings so that other front-end engineers join us in accelerating the user experience on the web.

The 80/20 Performance Rule

Vilfredo Pareto, an economist in the early 1900s, made a famous observation where 80% of the nation's wealth belonged to 20% of the population. This was later generalized into what's commonly referred to as the Pareto principle (also known as the 80-20 rule), which states for any phenomenon, 80% of the consequences come from 20% of the causes. We see this phenomenon in software engineering where 80% of the time is spent in only 20% of the code. When we optimize our applications, we know to focus on that 20% of the code. This same technique should also be applied when optimizing web pages. Most performance optimization today are made on the parts that generate the HTML document (apache, C++, databases, etc.) but these parts only

SYNDICATE
All Entries:
ADD MY YAHOO! RSS
All Comments:
ADD MY YAHOO! RSS

RECENT POSTS
Welcoming Mike Lee and Dav Glass to the YUI Team
YUI Theater – Doug Geoffray: "From the Mouth of a Screenreader"
YUI Implementation Focus: Dustin Diaz's DED|Chain
Happy Birthday to the Pattern Library, Too!
YUI Theater – "Browser Wars Episode II: Attack of the DOMs"
Free Excerpt: Nicholas Zakas on YUI Connection Manager, from Professional Ajax 2nd Edition

RECENT READERS

tenni08

80/20 Performance Rule

Vilfredo Pareto:

80% of consequences come from 20% of causes

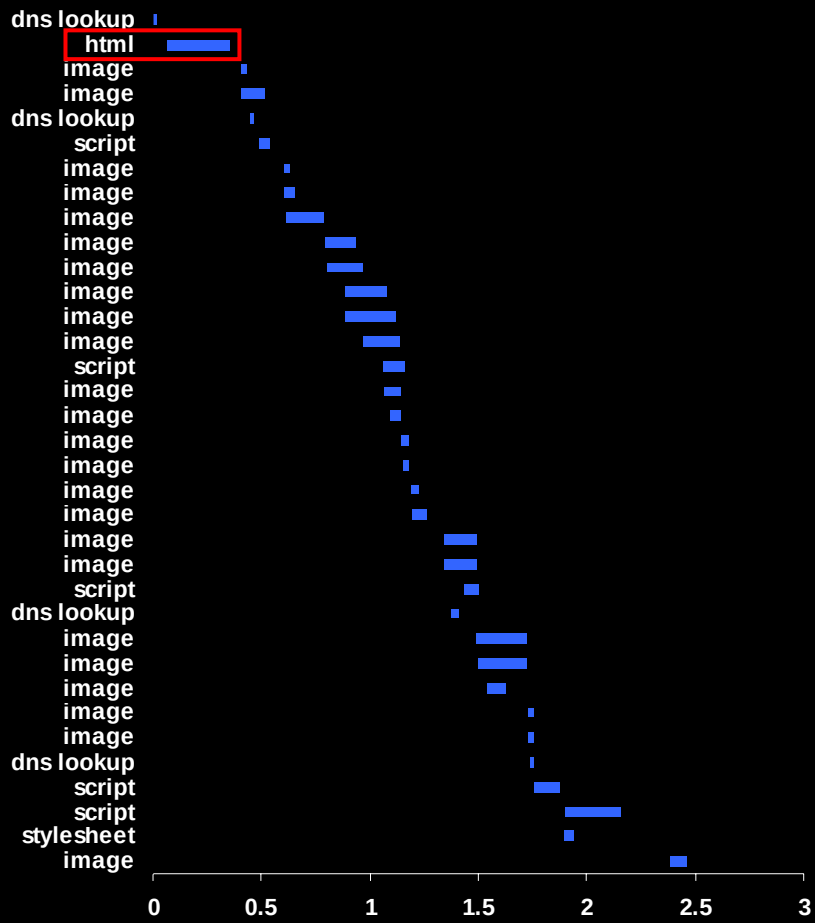
Focus on the 20% that affects 80% of the end-user response time.

Start at the front-end.

Empty vs. Full Cache

Empty vs. Full Cache

with an empty cache



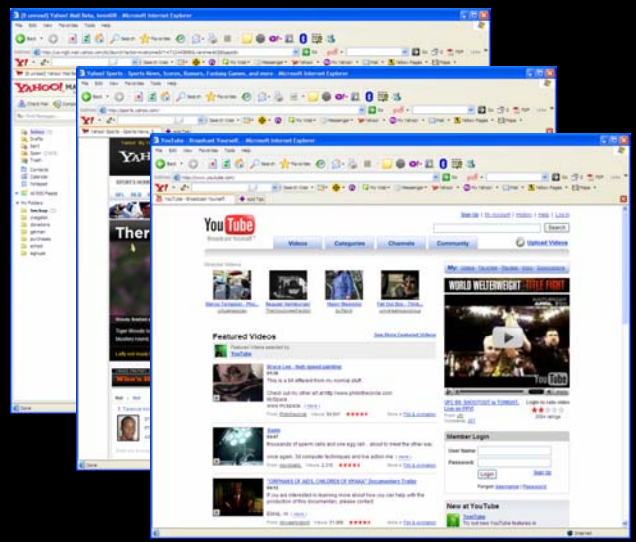
Empty vs. Full Cache

1

user requests
www.yahoo.com

2

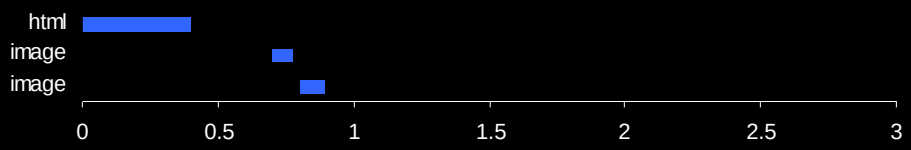
user requests
other web pages



3

user re-requests
www.yahoo.com

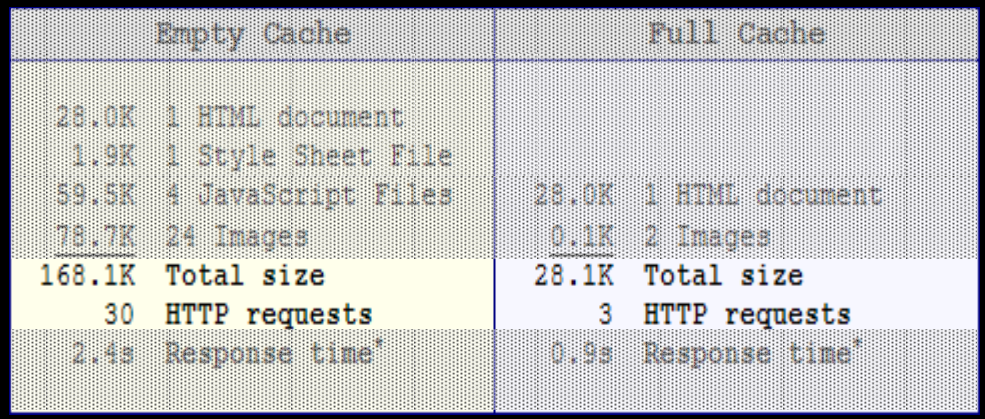
Empty vs. Full Cache



Expires header

with a full cache

Empty vs. Full Cache



Empty Cache		Full Cache	
28.0K	1 HTML document		
1.9K	1 Style Sheet File		
59.5K	4 JavaScript Files	28.0K	1 HTML document
78.7K	24 Images	0.1K	2 Images
168.1K	Total size	28.1K	Total size
30	HTTP requests	3	HTTP requests
2.4s	Response time*	0.9s	Response time*

empty cache

2.4 seconds

full cache

0.9 seconds

83% fewer bytes

90% fewer HTTP requests

How much does this benefit our users?

It depends on how many users have components in cache.

- What percentage of users view a page with an empty cache*?

  * "Empty cache" means the browser has to request the components instead of pulling them from the browser disk cache.

- What percentage of page views are done with an empty cache*?

Performance Research, Part 2: Browser Cache Usage - Exposed!

January 4, 2007 at 12:24 pm by Tenni Theurer | In Development |

This is the second in a series of articles describing experiments conducted to learn more about optimizing web page performance. You may be wondering why you're reading a performance article on the YUI Blog. It turns out that most of web page performance is affected by front-end engineering, that is, the user interface design and development.

In an earlier post, I described What the 80/20 Rule Tells Us about Reducing HTTP Requests. Since browsers spend 80% of the time fetching external components including scripts, stylesheets and images, reducing the number of HTTP requests has the biggest impact on reducing response time. But shouldn't everything be saved in the browser's cache anyway?

Why does cache matter?

It's important to differentiate between end user experiences for an empty versus a full cache page view. An "empty cache" means the browser bypasses the disk cache and has to request all the components to load the page. A "full cache" means all (or at least most) of the components are found in the disk cache and the corresponding HTTP requests are avoided.

The main reason for an empty cache page view is because the user is visiting the page for the first time and the browser has to download all the components to load the page. Other reasons include:

- The user visited the page previously but cleared the browser cache.
- The browser cache was automatically cleared, based on the browser's settings

SYNDICATE
All Entries:
MY YAHOO! RSS
All Comments:
MY YAHOO! RSS

RECENT POSTS
Welcoming Mike Lee and Dav Glass to the YUI Team
YUI Theater – Doug Geoffray: "From the Mouth of a Screenreader"
YUI Implementation Focus: Dustin Diaz's DED|Chain
Happy Birthday to the Pattern Library, Too!
YUI Theater – "Browser Wars Episode II: Attack of the DOMs"
Free Excerpt: Nicholas Zakas on YUI Connection Manager, from Professional Ajax 2nd Edition

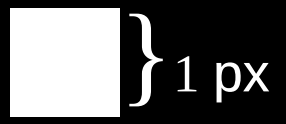
RECENT READERS

tenni08

Browser Cache Experiment

Add a new image to your page

```
<img src="image/blank.gif" height="1" width="1"/>
```



with the following response headers:

```
Expires: Thu, 15 Apr 2004 20:00:00 GMT
Last-Modified: Wed, 28 Sep 2006 23:49:57 GMT
```

Browser Cache Experiment

Requests from the browser will have one of these response status codes:

200 – The browser does not have the image in its cache.

304 – The browser has the image in its cache, but needs to verify the last modified date.

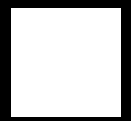
Browser Cache Experiment

What percentage of **users** view with an empty cache?

$$\frac{\text{\# unique users with at least one 200 response}}{\text{total \# unique users}}$$

What percentage of **page views** are done with an empty cache?

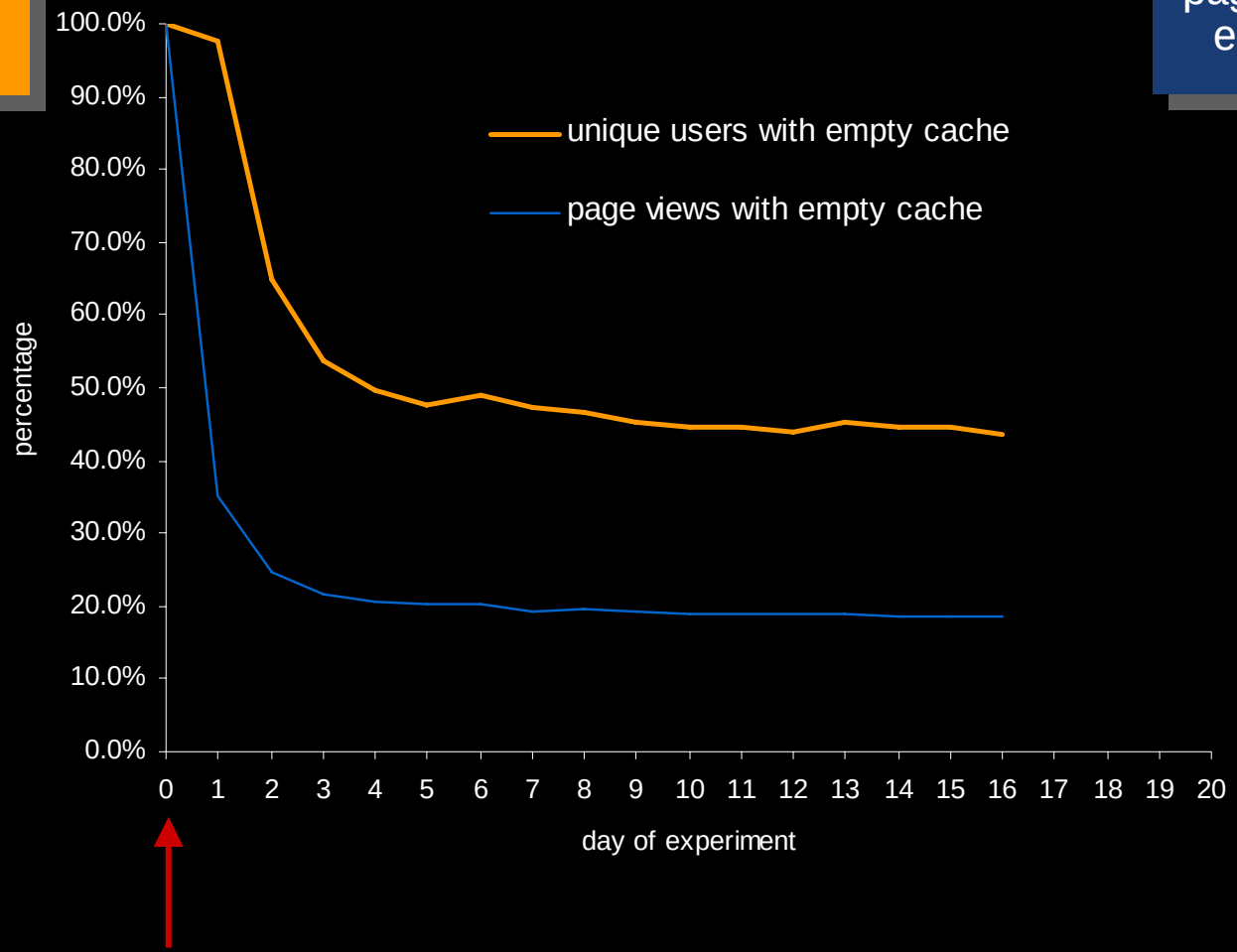
$$\frac{\text{total \# of 200 responses}}{\text{\# of 200 + \# of 304 responses}}$$



Surprising Results

users with empty cache

40-60%



page views with empty cache

~20%

Experiment Takeaways

Keep in mind the empty cache user experience. It might be more prevalent than you think!

Use different techniques to optimize full versus empty cache experience.

Performance Research, Part 3: When the Cookie Crumbles

March 1, 2007 at 4:41 pm by Tenni Theurer | In Development |

This article, co-written by Patty Chi, is the third in a series of articles describing experiments conducted to learn more about optimizing web page performance (Part 1, Part 2). You may be wondering why you're reading a performance article on the YUI Blog. It turns out that most of web page performance is affected by front-end engineering — that is, the user interface design and development.

HTTP cookies are used for a variety of reasons such as authentication and personalization. Information about cookies is exchanged in the HTTP headers between web servers and browsers. This article discusses the impact of cookies on the overall user response time.

HTTP Quick Review

Cookies originate from web servers when browsers request a page. Here is a sample HTTP header sent by the web server after a request for `www.yahoo.com`:

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Set-Cookie: C=abcde; path=/; domain=.yahoo.com
```

The header includes information about the response such as the protocol version, status code, and content-type. The `Set-Cookie` is also included in the response and in this example the name of the cookie is "C" and the value of the cookie is "abcde". Note: The maximum size of a cookie is

SYNDICATE
All Entries:
ADD MY YAHOO! RSS
All Comments:
ADD MY YAHOO! RSS

RECENT POSTS
Welcoming Mike Lee and Dav Glass to the YUI Team
YUI Theater – Doug Geoffray: "From the Mouth of a Screenreader"
YUI Implementation Focus: Dustin Diaz's DED|Chain
Happy Birthday to the Pattern Library, Too!
YUI Theater – "Browser Wars Episode II: Attack of the DOMs"
Free Excerpt: Nicholas Zakas on YUI Connection Manager, from Professional Ajax 2nd Edition

RECENT READERS

tenni08

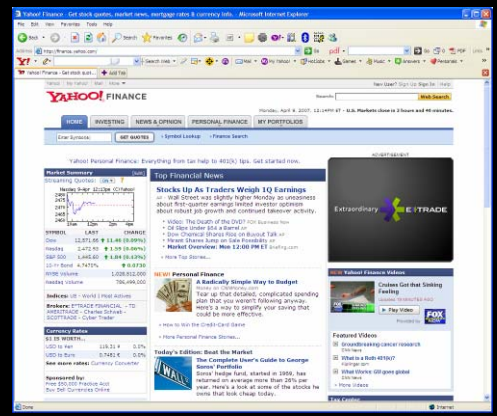
HTTP Quick Review

user requests
www.yahoo.com

HTTP response header sent by the web server:

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Set-Cookie: C=abcdefghijklmnopqrstuvwxyz; domain=.yahoo.com
```

HTTP Quick Review



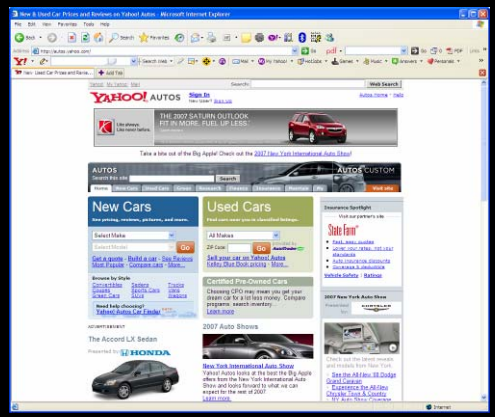
HTTP request header sent by the browser:

```
GET / HTTP/1.1
Host: finance.yahoo.com
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; …
Cookie: C=abcdefghijklmnopqrstuvwxyz;
```

HTTP Quick Review

user requests
www.yahoo.com

user requests
autos.yahoo.com



HTTP request header sent by the browser:

```
GET / HTTP/1.1
Host: autos.yahoo.com
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; …
Cookie: C=abcdefghijklmnopqrstuvwxyz;
```

HTTP Quick Review

HTTP request header sent by the browser:

```
GET / HTTP/1.1
Host: mail.yahoo.com
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; …
Cookie: C=abcdefghijklmnopqrstuvwxyz;
```

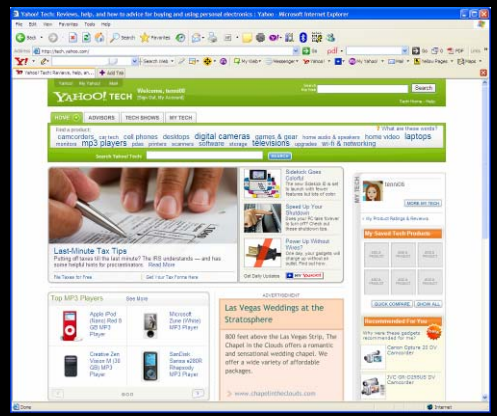
HTTP Quick Review

1

user requests
www.yahoo.com

5

user requests
tech.yahoo.com



HTTP request header sent by the browser:

```
GET / HTTP/1.1
Host: tech.yahoo.com
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; …
Cookie: C=abcdefghijklmnopqrstuvwxyz;
```

Impact of Cookies on Response Time

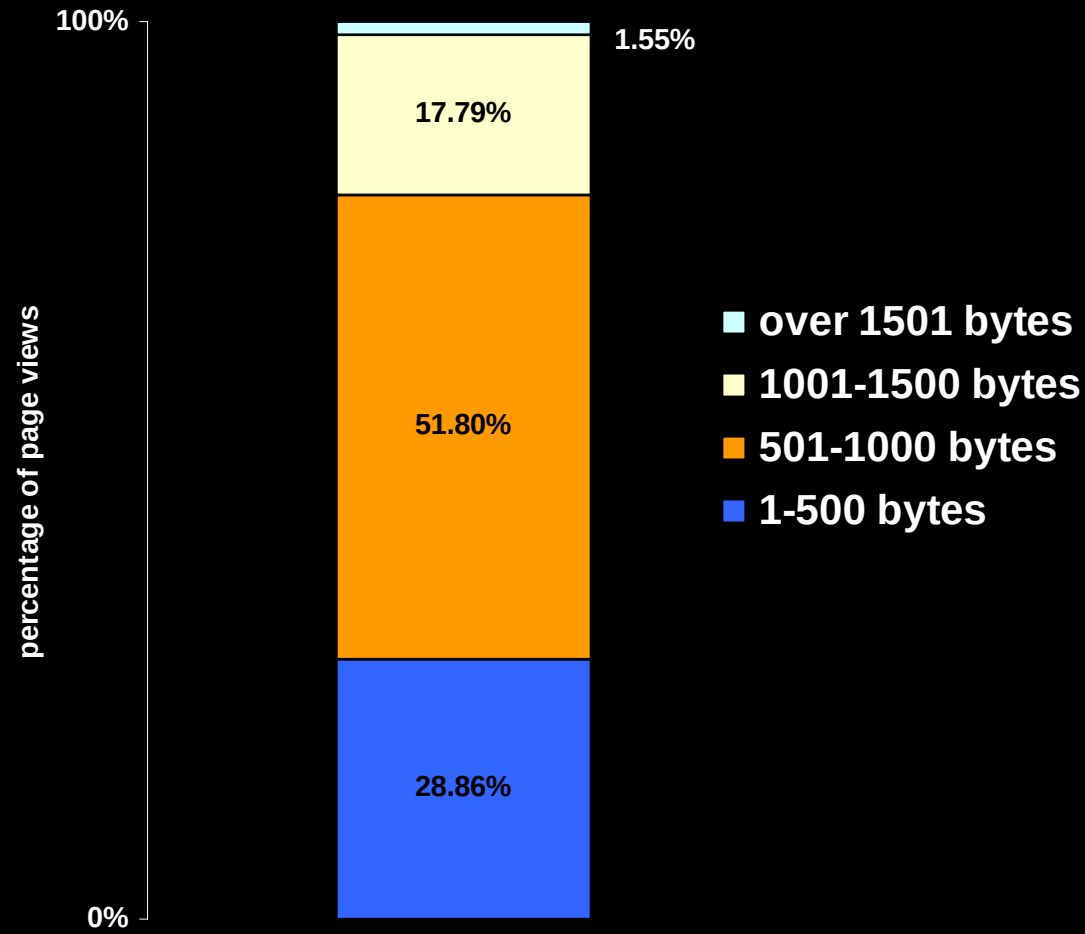
Cookie Size	Time	Delta
0 bytes	78 ms	0 ms
500 bytes	79 ms	+1 ms
1000 bytes	94 ms	+16 ms
1500 bytes	109 ms	+31 ms
2000 bytes	125 ms	+47 ms
2500 bytes	141 ms	+63 ms
3000 bytes	156 ms	+78 ms

keep sizes low

80 ms delay

dialup users

.yahoo.com cookie sizes



Analysis of Cookie Sizes across the Web

	Total Cookie Size
Amazon	60 bytes
Google	72 bytes
Yahoo	122 bytes
CNN	184 bytes
YouTube	218 bytes
MSN	268 bytes
eBay	331 bytes
MySpace	500 bytes

Experiment Takeaways

eliminate unnecessary cookies

keep cookie sizes low

set cookies at appropriate domain level

set Expires date appropriately

- earlier date or none removes cookie sooner

Performance Research, Part 4: Maximizing Parallel Downloads in the Carpool Lane

April 11, 2007 at 11:47 am by Tenni Theurer | In Development |

This article, co-written by Steve Souders, is the fourth in a series of articles describing experiments conducted to learn more about optimizing web page performance (Part 1, Part 2, Part 3). You may be wondering why you're reading a performance article on the YUI Blog. It turns out that most of web page performance is affected by front-end engineering, that is, the user interface design and development.

Parallel Downloads

The biggest impact on end-user response times is the number of components in the page. Each component requires an extra HTTP request, perhaps not when the cache is full, but definitely when the cache is empty. Knowing that the browser performs HTTP requests in parallel, you may ask why the number of HTTP requests affects response time. Can't the browser download them all at once?

The explanation goes back to the HTTP/1.1 spec, which suggests that browsers download two components in parallel per hostname. Many web pages download all their components from a single hostname. Viewing these HTTP requests reveals a stair-step pattern, as shown

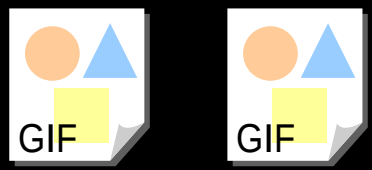
SYNDICATE
All Entries:
MY YAHOO! RSS
All Comments:
MY YAHOO! RSS

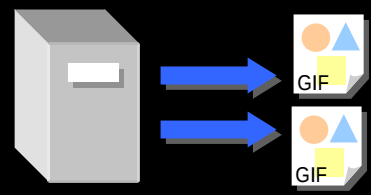
RECENT POSTS
Performance Research, Part 4: Maximizing Parallel Downloads in the Carpool Lane
JSON and Browser Security
YUI Version 2.2.1 Released, and Graded Browser Support Update
Welcoming Mike Lee and Dav Glass to the YUI Team
YUI Theater – Doug Geoffray: "From the Mouth of a Screenreader"
YUI Implementation Focus: Dustin Diaz's DED|Chain

RECENT READERS

Parallel Downloads

Two components in parallel per hostname





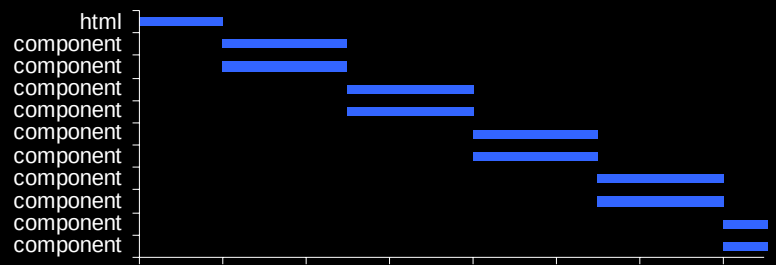
HTTP/1.1

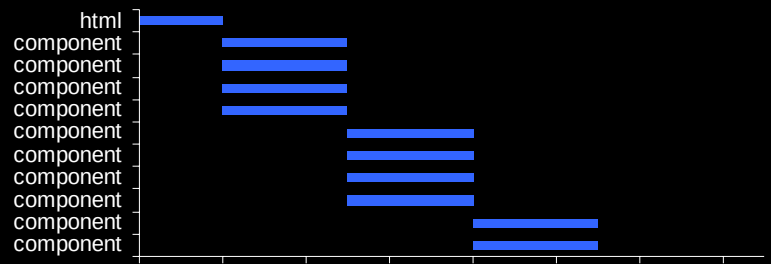
Parallel Downloads

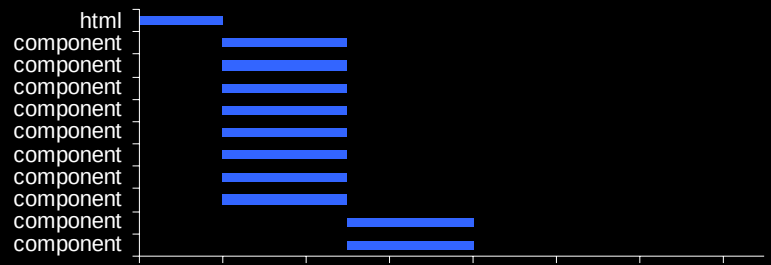
Two in parallel

Four in parallel

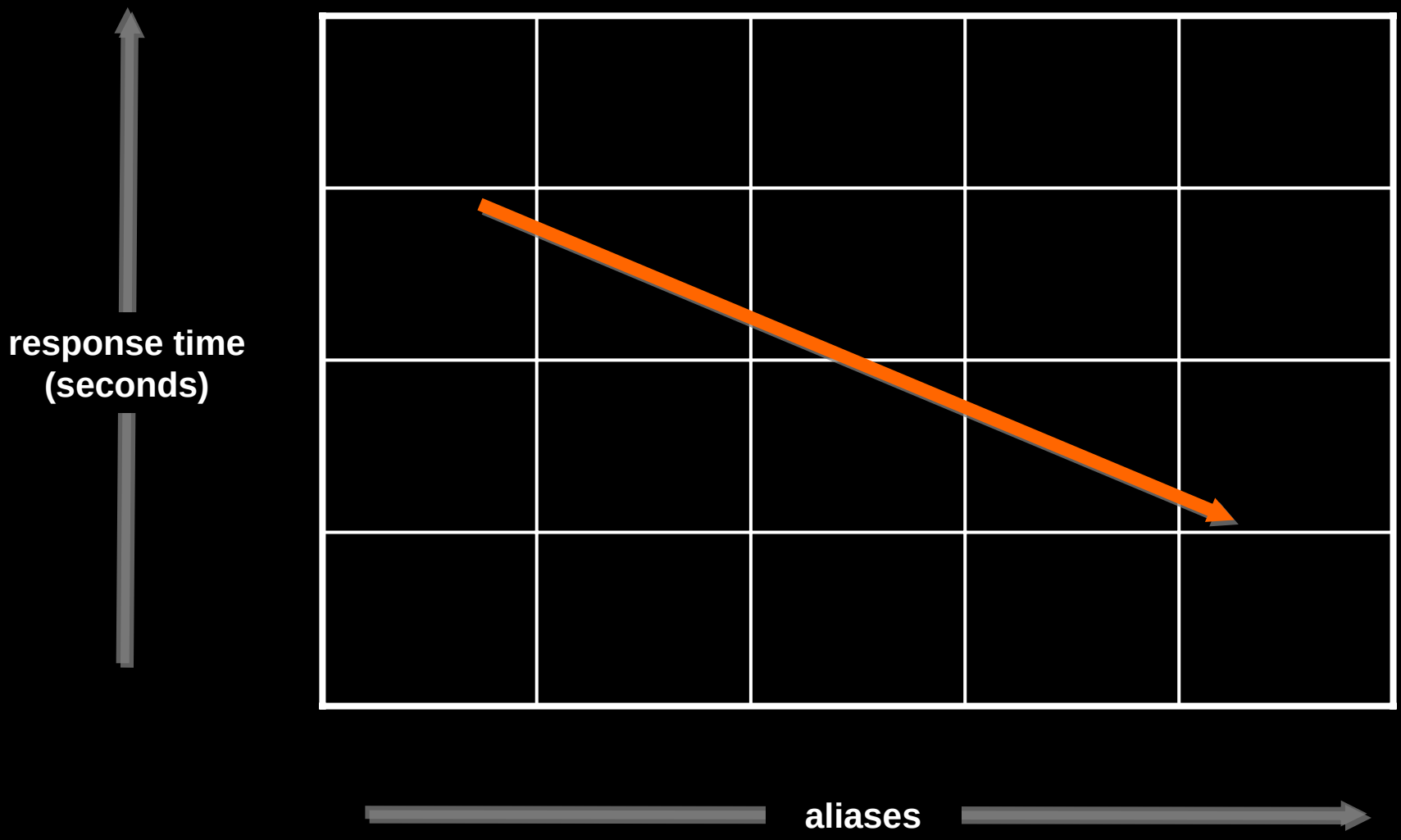
Eight in parallel



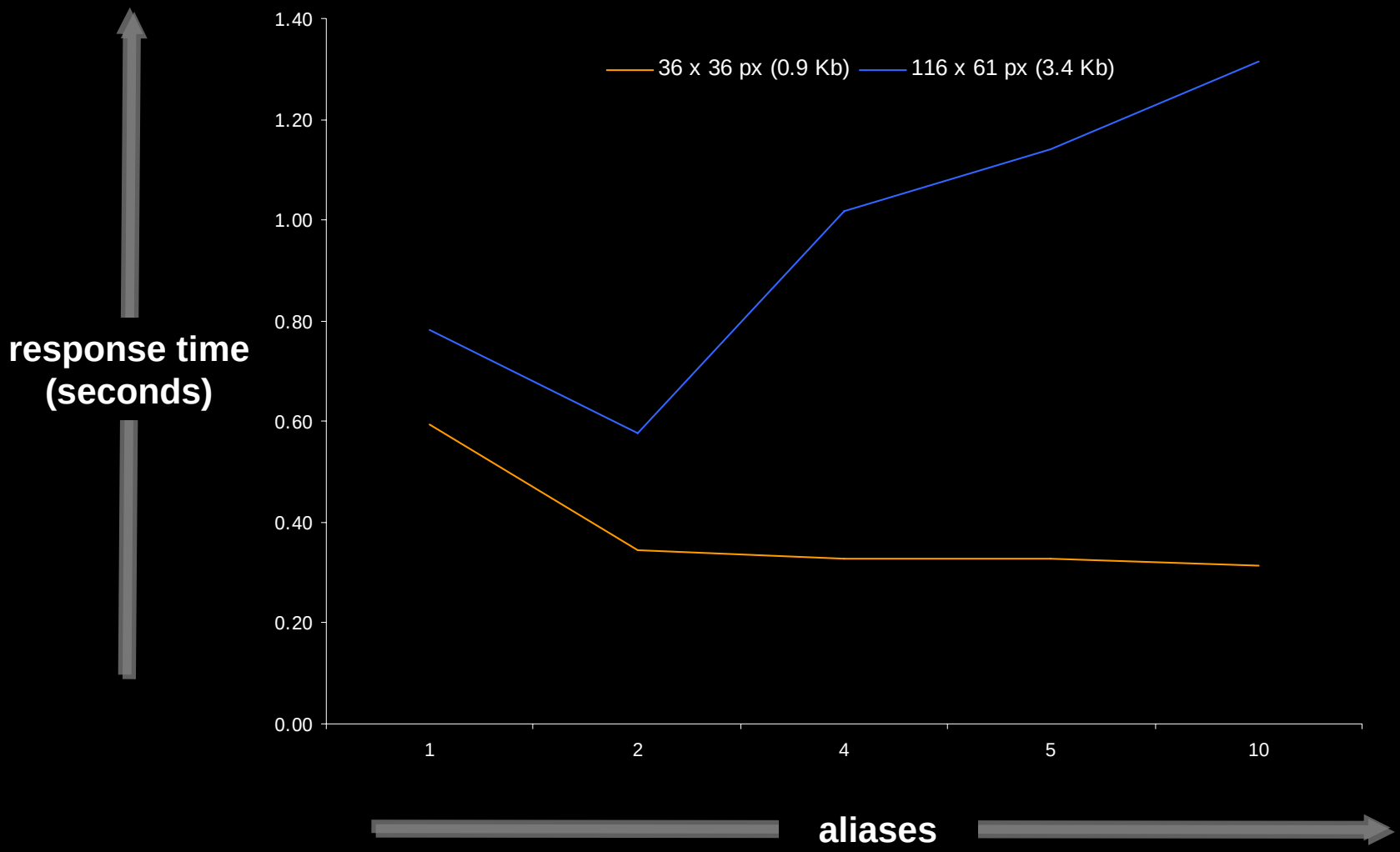




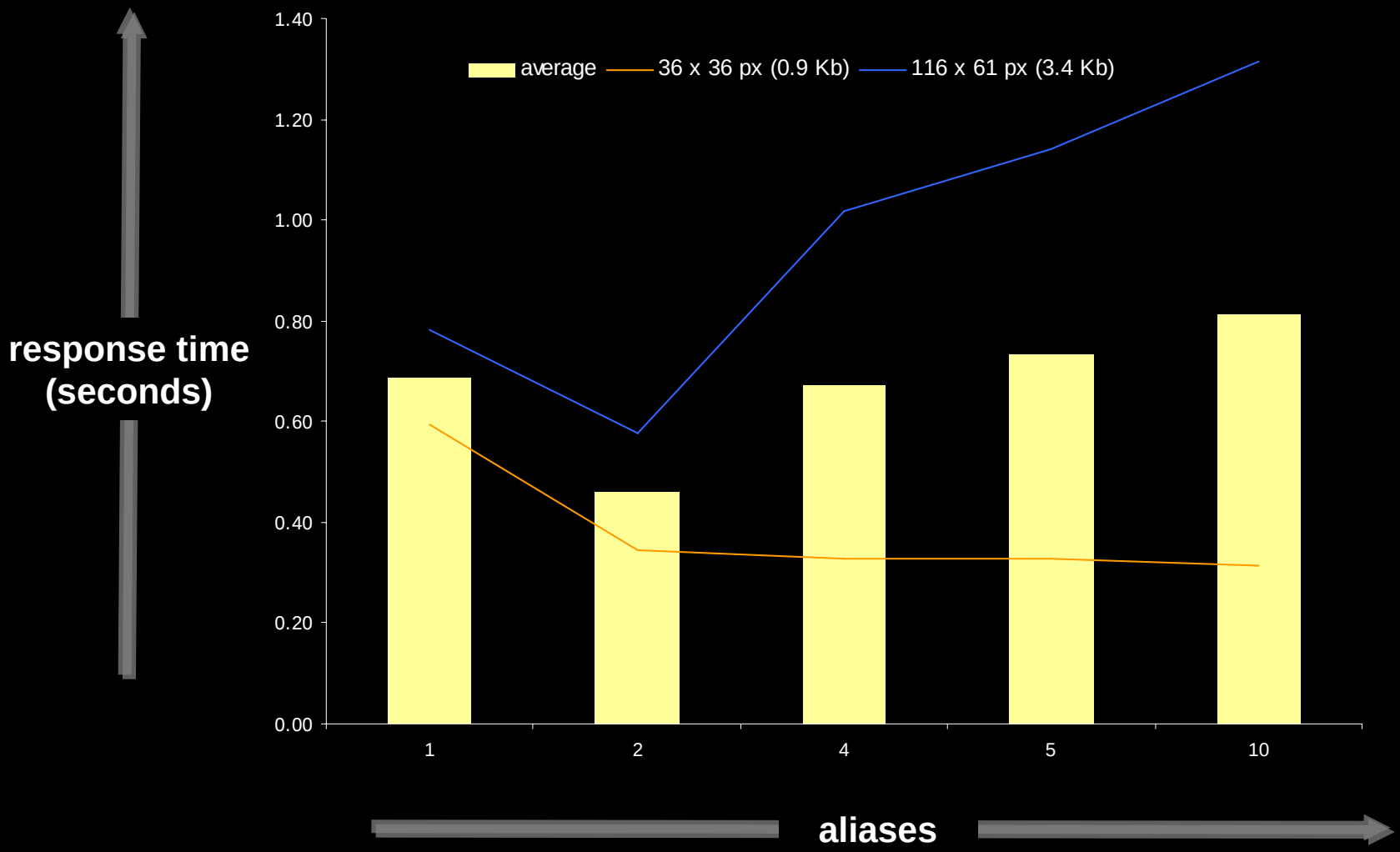
Maximizing Parallel Downloads



Maximizing Parallel Downloads

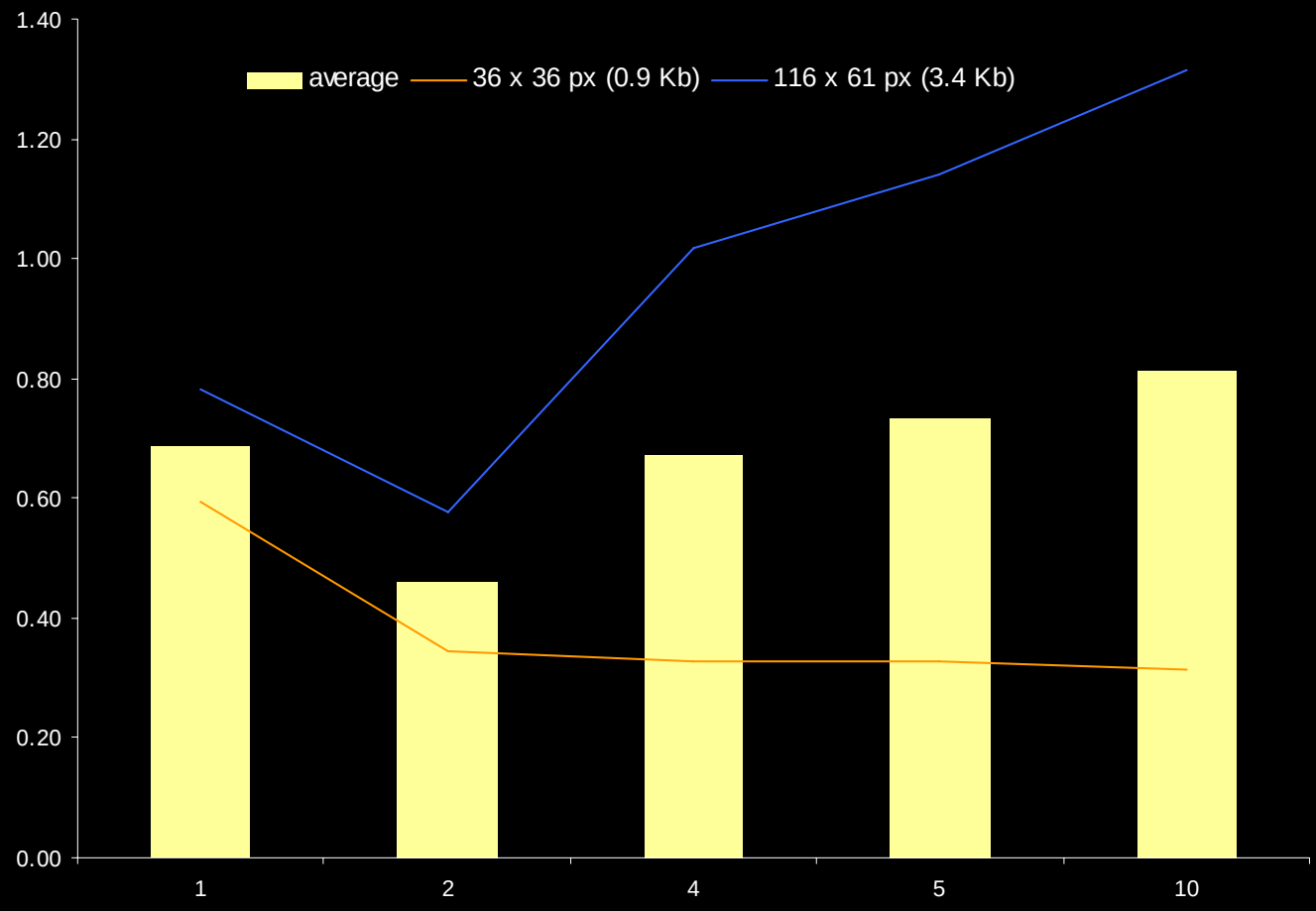


Maximizing Parallel Downloads



Maximizing Parallel Downloads

response time (seconds)



rule of thumb: use at least two but no more than four aliases

Experiment Takeaways

- consider the effects of CPU thrashing
- DNS lookup times vary across ISPs and geographic locations
- domain names may not be cached

Summary

- What the 80/20 Rule Tells Us about Reducing HTTP Requests
  http://yuiblog.com/blog/2007/04/11/performance-research-part-4/
- Browser Cache Usage – Exposed!
  http://yuiblog.com/blog/2007/01/04/performance-research-part-2/
- When the Cookie Crumbles
  http://yuiblog.com/blog/2007/01/04/performance-research-part-2/
- Maximizing Parallel Downloads in the Carpool Lane
  http://yuiblog.com/blog/2007/04/11/performance-research-part-4/

14 Rules

14 Rules

1. Make fewer HTTP requests
2. Use a CDN
3. Add an Expires header
4. Gzip components
5. Put CSS at the top
6. Move JS to the bottom
7. Avoid CSS expressions
8. Make JS and CSS external
9. Reduce DNS lookups
10. Minify JS
11. Avoid redirects
12. Remove duplicate scripts
13. Turn off ETags
14. Make AJAX cacheable and small

Rule 1: Make fewer HTTP requests

- image maps
- CSS sprites
- inline images
- combined scripts, combined stylesheets

Image maps

server-side

```
<a href="navbar.cgi"><img ismap src="imagemap.gif"></a>
→ http://.../navbar.cgi?127,13
```

client-side – preferred

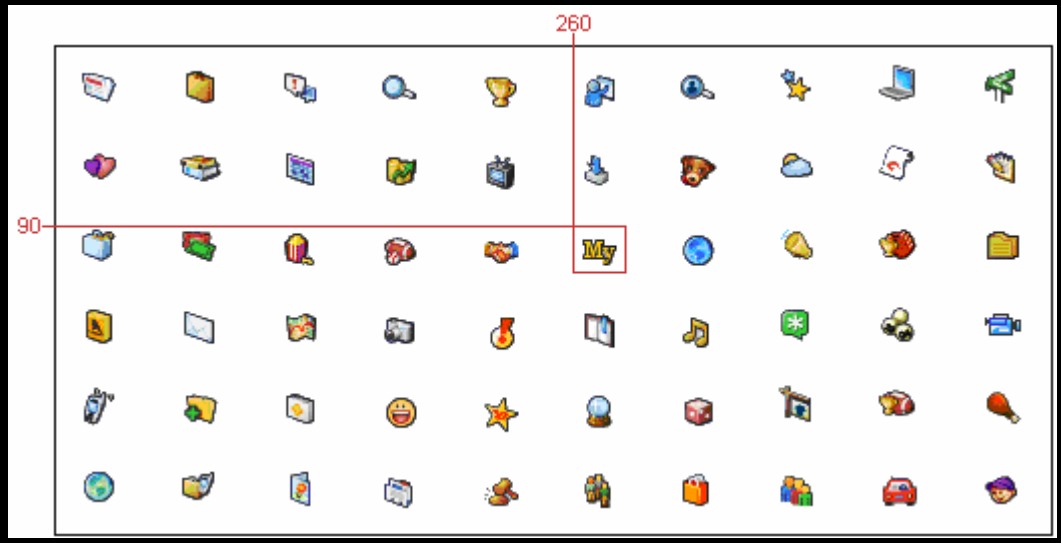
```
<img usemap="#map1" border=0 src="/images/imagemap.gif">
<map name="map1">
  <area shape="rect" coords="0,0,31,31" href="home.html" title="Home">
   …
</map>
```

drawbacks:

- must be contiguous
- defining area coordinates – tedious, errors

http://www.w3.org/TR/html401/struct/objects.html#h-13.6

CSS Sprites - Preferred




```
<span style="
  background-image: url('sprites.gif');
  background-position: -260px -90px;">
</span>
```

size of combined image is less

not supported in Opera 6

http://alistapart.com/articles/sprites

Inline Images

`data:` URL scheme

```
data:[<mediatype>][;base64],<data>
```

```
<IMG ALT=”Red Star” 
SRC="
   ajo+w6O/zl5estLv/8/AAAAAAAAAAAAAAACH5BAEAAAsALAAAAAAMAAwAAAQzcElZyryT
   EHyTUgknHd9xGV+qKsYirKkwDYiKDBiatt2H1KBLQRFIJAIKywRgmhwAIlEEADs=">
```

<u>not supported in IE</u>

avoid increasing size of HTML pages:

- put inline images in cached stylesheets

http://tools.ietf.org/html/rfc2397

Combined Scripts, Combined Stylesheets

	Scripts	Stylesheets
amazon.com	3	1
aol.com	18	1
cnn.com	11	2
ebay.com	7	2
froogle.google.com	1	1
msn.com	9	1
myspace.com	2	2
wikipedia.org	3	1
yahoo.com	4	1
youtube.com	7	3
Average	**6.5**	**1.5**

Combined Scripts, Combined Stylesheets

combining six scripts into one eliminates five HTTP requests

challenges:

- develop as separate modules
- number of possible combinations vs. loading more than needed
- maximize browser cache

one solution:

- dynamically combine and cache

Rule 2: Use a CDN

amazon.com	Akamai
aol.com	Akamai
cnn.com	
ebay.com	Akamai, Mirror Image
google.com	
msn.com	SAVVIS
myspace.com	Akamai, Limelight
wikipedia.org	
yahoo.com	Akamai
youtube.com	

distribute your static content before distributing your dynamic content

Rule 3: Add an Expires header

not just for images

	Images	**Stylesheets**	**Scripts**	%	Median Age
amazon.com	0/62	0/1	0/3	0%	114 days
aol.com	23/43	1/1	6/18	48%	217 days
cnn.com	0/138	0/2	2/11	1%	227 days
ebay.com	16/20	0/2	0/7	55%	140 days
froogle.google.com	1/23	0/1	0/1	4%	454 days
msn.com	32/35	1/1	3/9	80%	34 days
myspace.com	0/18	0/2	0/2	0%	1 day
wikipedia.org	6/8	1/1	2/3	75%	1 day
yahoo.com	23/23	1/1	4/4	100%	n/a
youtube.com	0/32	0/3	0/7	0%	26 days

Rule 4: Gzip components

you can affect users' download times

90%+ of browsers support compression

Gzip vs. Deflate

		Gzip		Deflate	
	Size	Size	Savings	Size	Savings
Script	3.3K	1.1K	67%	1.1K	66%
Script	39.7K	14.5K	64%	16.6K	58%
Stylesheet	1.0K	0.4K	56%	0.5K	52%
Stylesheet	14.1K	3.7K	73%	4.7K	67%

Gzip compresses more

Gzip supported in more browsers

Gzip: not just for HTML

	HTML	Scripts	Stylesheets
amazon.com	x		
aol.com	x	some	some
cnn.com			
ebay.com	x		
froogle.google.com	x	x	x
msn.com	x	deflate	deflate
myspace.com	x	x	x
wikipedia.org	x	x	x
yahoo.com	x	x	x
youtube.com	x	some	some

gzip scripts, stylesheets, XML, JSON (not images, PDF)

Gzip Configuration

Apache 2.x: mod_deflate

```
AddOutputFilterByType DEFLATE text/html text/css
  application/x-javascript
```

HTTP request

```
Accept-Encoding: gzip, deflate
```

HTTP response

```
Content-Encoding: gzip
Vary: Accept-Encoding
```

needed for proxies

Gzip Edge Cases

<1% of browsers have problems with gzip

- IE 5.5:
  http://support.microsoft.com/default.aspx?scid=kb;en-us;Q313712
- IE 6.0:
  http://support.microsoft.com/default.aspx?scid=kb;en-us;Q31249
- Netscape 3.x, 4.x
  http://www.schroepl.net/projekte/mod_gzip/browser.htm

consider adding `Cache-Control: Private`

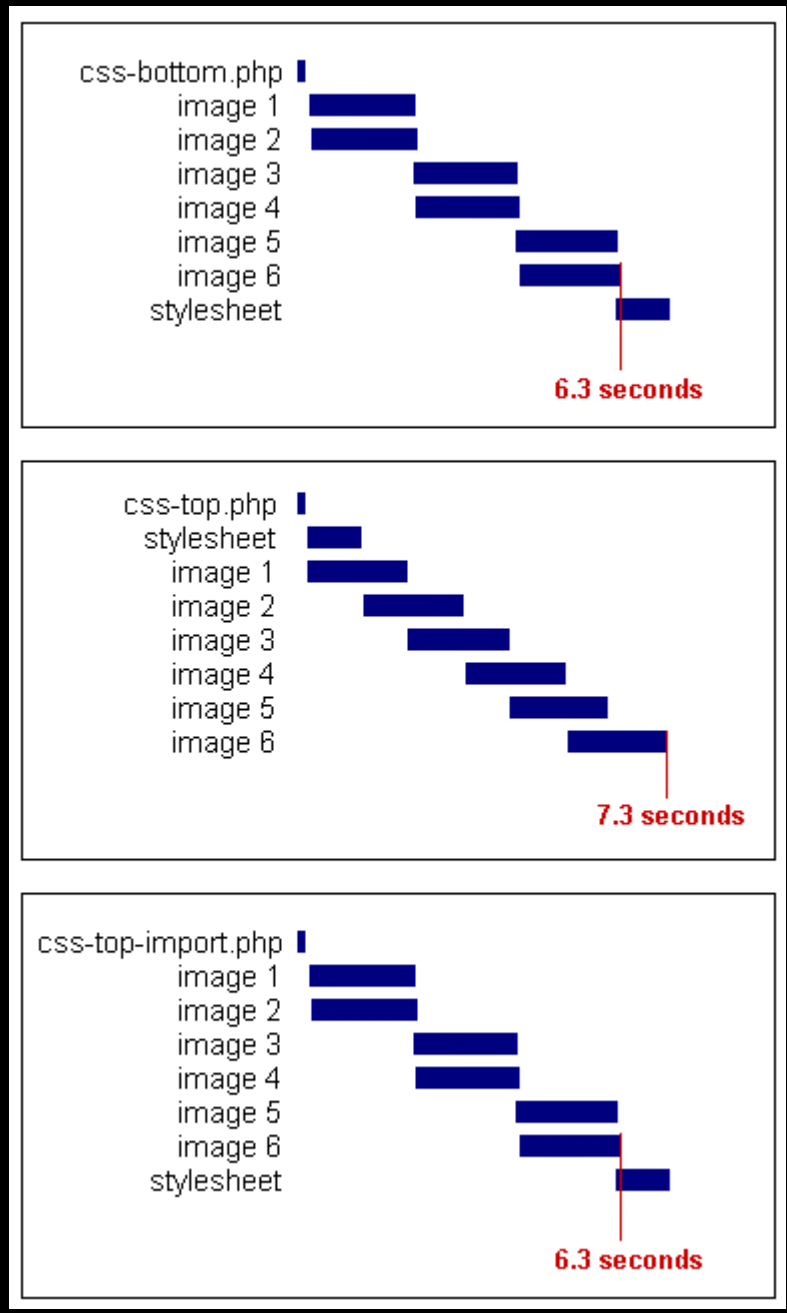
remove ETags (Rule 13)

hard to diagnose; problem getting smaller

Rule 5: Put CSS at the top

- stylesheets block rendering in IE
  - http://stevesouders.com/examples/css-bottom.php
- solution: put stylesheets in HEAD (per spec)
- avoids Flash of Unstyled Content
- use LINK (not @import)

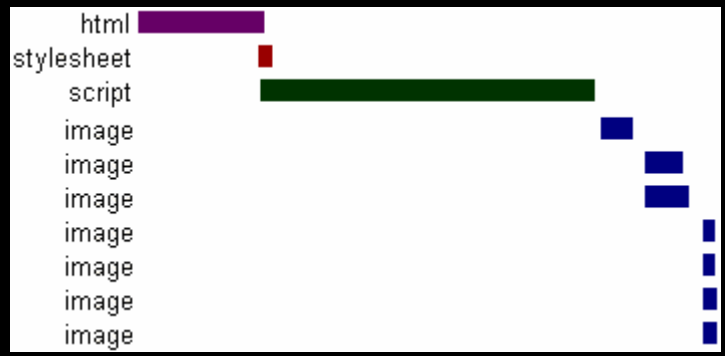
Slowest is Fastest



Rule 6: Move scripts to the bottom

- scripts block parallel downloads across all hostnames
- scripts block rendering of everything below them in the page
- IE and FF

  http://stevesouders.com/examples/js-middle.php



Rule 6: Move scripts to the bottom

- `script defer` attribute is not a solution
  - blocks rendering and downloads in FF
  - slight blocking in IE
- solution: move them as low in the page as possible

Rule 7: Avoid CSS expressions

used to set CSS properties dynamically in IE

```
width: expression(
  document.body.clientWidth < 600 ?
  “600px” : “auto” );
```

problem: expressions execute many times

- mouse move, key press, resize, scroll, etc.

http://stevesouders.com/examples/expression-counter.php

One-Time Expressions

expression overwrites itself

```
<style>
P {
    background-color: expression(altBgcolor(this));
}
</style>

<script>
function altBgcolor(elem) {
    elem.style.backgroundColor = (new
  Date()).getHours()%2 ? "#F08A00" : "#B8D4FF";
}
</script>
```

Event Handlers

tie behavior to (fewer) specific events

```
window.onresize = setMinWidth;
function setMinWidth() {
    var aElements =
      document.getElementsByTagName("p");
    for ( var i = 0; i < aElements.length; i++ ) {
        aElements[i].runtimeStyle.width = (
           document.body.clientWidth<600 ?
           "600px" : "auto" );
    }
}
```

Rule 8: Make JS and CSS external

inline: HTML document is bigger

external: more HTTP requests, but cached

variables

- page views per user (per session)
- empty vs. full cache stats
- component re-use

external is typically better

- home pages may be an exception

Post-Onload Download

inline in front page

download external files after onload

```
window.onload = downloadComponents;
function downloadComponents() {
    var elem = document.createElement("script");
    elem.src = "http://.../file1.js";
    document.body.appendChild(elem);
    ...
}
```

speeds up secondary pages

Dynamic Inlining

start with post-onload download

set cookie after components downloaded

server-side:

- if cookie, use external
- else, do inline with post-onload download

cookie expiration date is key

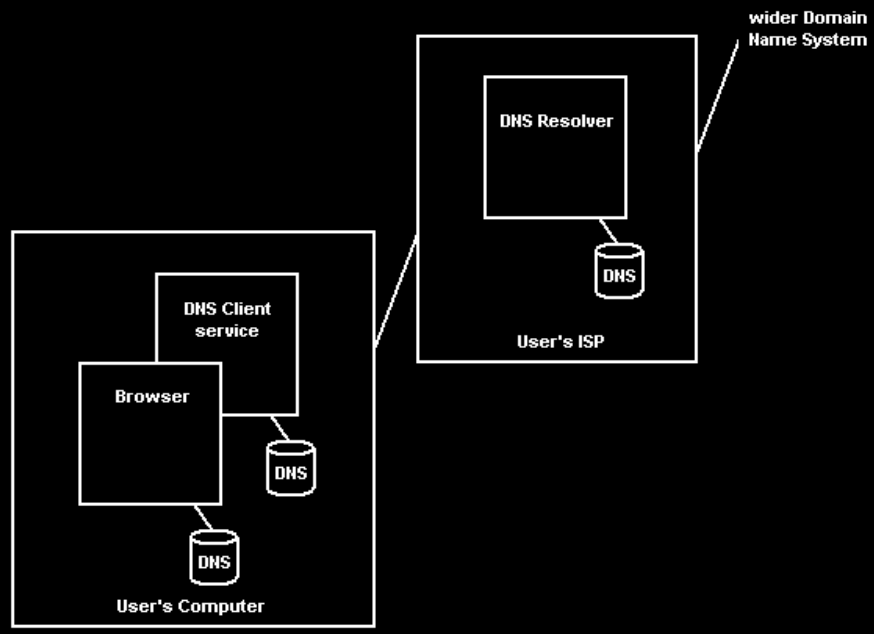
speeds up all pages

Rule 9: Reduce DNS lookups

typically 20-120 ms

block parallel downloads

OS and browser both have DNS caches



TTL (Time To Live)

www.amazon.com	1 minute
www.aol.com	1 minute
www.cnn.com	10 minutes
www.ebay.com	1 hour
www.google.com	5 minutes
www.msn.com	5 minutes
www.myspace.com	1 hour
www.wikipedia.org	1 hour
www.yahoo.com	1 minute
www.youtube.com	5 minutes

TTL – how long record can be cached

browser settings override TTL

Browser DNS Cache

IE

- DnsCacheTimeout: 30 minutes
- KeepAliveTimeout: 1 minute
- ServerInfoTimeout: 2 minutes

Firefox

- network.dnsCacheExpiration: 1 minute
- network.dnsCacheEntries: 20
- network.http.keep-alive.timeout: 5 minutes
- Fasterfox: 1 hour, 512 entries, 30 seconds

Reducing DNS Lookups

fewer hostnames – 2-4

keep-alive

Rule 10: Minify JavaScript

	Minify External?	Minify Inline?
www.amazon.com	no	no
www.aol.com	no	no
www.cnn.com	no	no
www.ebay.com	yes	no
froogle.google.com	yes	yes
www.msn.com	yes	yes
www.myspace.com	no	no
www.wikipedia.org	no	no
www.yahoo.com	yes	yes
www.youtube.com	no	no

minify inline scripts, too

Minify vs. Obfuscate

	Original	JSMin Savings	Dojo Savings
www.amazon.com	204K	31K (15%)	48K (24%)
www.aol.com	44K	4K (10%)	4K (10%)
www.cnn.com	98K	19K (20%)	24K (25%)
www.myspace.com	88K	23K (27%)	24K (28%)
www.wikipedia.org	42K	14K (34%)	16K (38%)
www.youtube.com	34K	8K (22%)	10K (29%)
Average	**85K**	**17K (21%)**	**21K (25%)**

minify – it's safer

http://crockford.com/javascript/jsmin

http://dojotoolkit.org/docs/shrinksafe

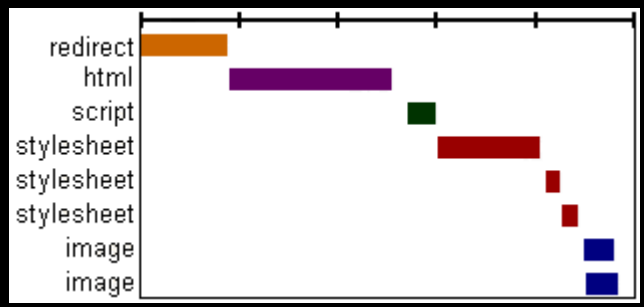
Rule 11: Avoid redirects

3xx status codes – mostly 301 and 302

```
HTTP/1.1 301 Moved Permanently
Location: http://stevesouders.com/newuri
```

add Expires headers to cache redirects

worst form of blocking



http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html

Redirects

	Redirects
www.amazon.com	no
www.aol.com	yes – secondary page
www.cnn.com	yes – initial page
www.ebay.com	yes – secondary page
froogle.google.com	no
www.msn.com	yes – initial page
www.myspace.com	yes – secondary page
www.wikipedia.org	yes – secondary page
www.yahoo.com	yes – secondary page
www.youtube.com	no

Avoid Redirects

missing trailing slash
- http://astrology.yahoo.com/astrology
- use `Alias` or `DirectorySlash`

`mod_rewrite`

CNAMEs

log referer – track internal links

outbound links – harder
- beacons – beware of race condition
- XHR – bail at readyState 2

Rule 12: Remove duplicate scripts

- hurts performance
  - extra HTTP requests (IE only)
  - extra executions
- atypical?
  - 2 of 10 top sites contain duplicate scripts
- team size, # of scripts

Script Insertion Functions

```
<?php
function insertScript($jsfile) {
    if ( alreadyInserted($jsfile) ) { return; }

    pushInserted($jsfile);

    if ( hasDependencies($jsfile) ) {
        $dependencies = getDependencies($jsfile);
        for ( $i = 0; $i < count($dependencies); $i++ ) {
            insertScript($dependencies[$i]);
        }
    }

    echo '<script type="text/javascript" src="' .
         getVersion($jsfile) . '"></script>";
}
?>
```

Rule 13: Turn off ETags

unique identifier returned in response

```
ETag: "c8897e-aee-4165acf0"
Last-Modified: Thu, 07 Oct 2004 20:54:08 GMT
```

used in conditional GET requests

```
If-None-Match: "c8897e-aee-4165acf0"
If-Modified-Since: Thu, 07 Oct 2004 20:54:08 GMT
```

if ETag doesn't match, can't send 304

The Problem with ETags

- ETag for a single entity is always different across servers
- ETag format
  - Apache: `inode-size-timestamp`
  - IIS: `Filetimestamp:ChangeNumber`
- Sites with >1 server return too few 304s
  - (n-1)/n
- Remove them
  - Apache: `FileETag none`
  - IIS: http://support.microsoft.com/kb/922703/

Rule 14: Make AJAX cacheable and small

- XHR, JSON, iframe, dynamic scripts can still be cached, minified, and gzipped
- a personalized response should still be cacheable by that person

AJAX Example: Yahoo! Mail Beta

address book XML request

```
→ GET /yab/[...]&r=0.5289571053069156 HTTP/1.1
  Host: us.xxx.mail.yahoo.com
← HTTP/1.1 200 OK
  Date: Thu, 12 Apr 2007 19:39:09 GMT
  Cache-Control: private,max-age=0
  Last-Modified: Sat, 31 Mar 2007 01:17:17 GMT
  Content-Type: text/xml; charset=utf-8
  Content-Encoding: gzip
```

address book changes infrequently

- cache it; add last-modified-time in URL

Live Analysis

IBM Page Detailer

packet sniffer

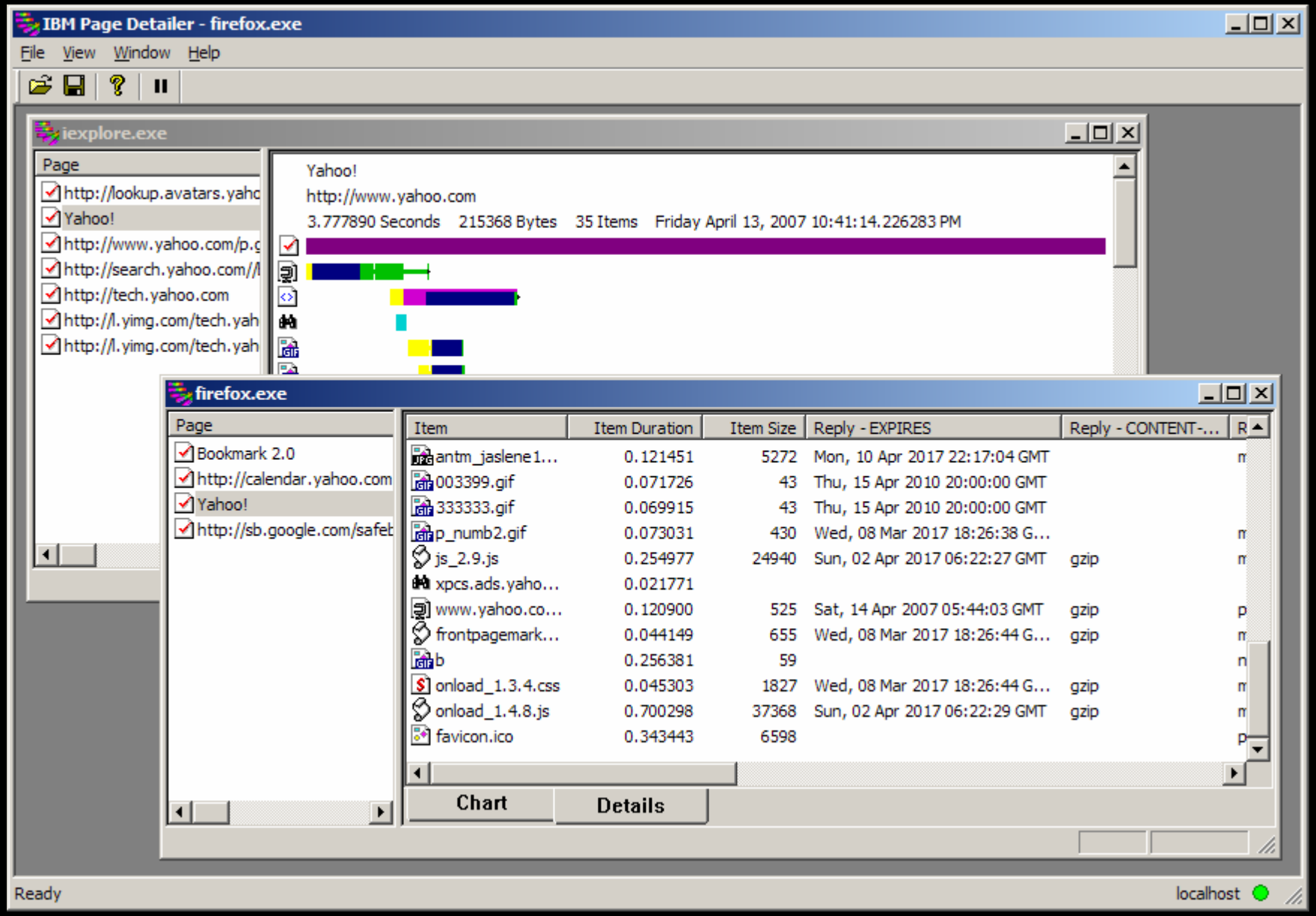
Windows only

IE, FF, any .exe

```
c:\windows\wd_WS2s.ini
Executable=(NETSCAPE.EXE),(NETSCP6.EXE),(firefox.exe)
```

free trial, $300 license

http://alphaworks.ibm.com/tech/pagedetailer



http://alphaworks.ibm.com/tech/pagedetailer

Fasterfox

measures load time of pages

alters config settings for faster loading

Firefox extension

free

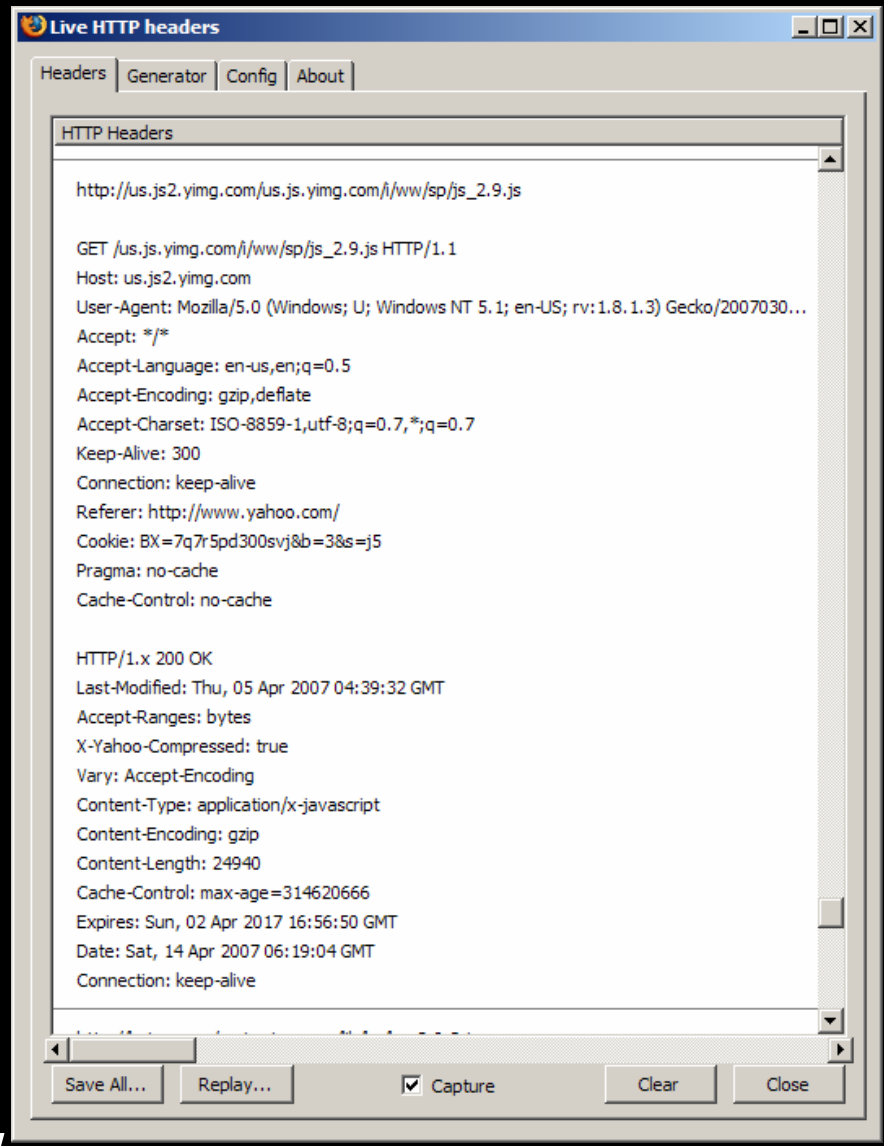
http://fasterfox.mozdev.org/

LiveHTTPHeaders

view HTTP headers

Firefox extension

free



http://livehttpheaders.mozdev.org/

Firebug

web development evolved

inspect and edit HTML

tweak and visualize CSS

debug and profile JavaScript

monitor network activity (caveat)

Firefox extension

free

http://getfirebug.com/

Yahoo! - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://www.yahoo.com/

Make Y! your home page

Yahoo! Voice: Make international calls for as low as 1¢/min.

at&t YAHOO!

HIGH SPEED INTERNET

My Yahoo! My Mail

Web | Images | Video | Local | Shopping | more

Search: Web Search

Y! Answers: Ask a question | Answer questions

Page Options

Autos
Calendar
Finance
Games
GeoCities
Groups
HotJobs
Maps
Movies
Music

Featured | Entertainment | Sports | Life

Apr 14, 2007

Disturbing thoughts

Find out what Yahoo! users think of "Disturbia" – a modern twist on the classic "Rear Window." » Reviews

• Showtimes Trailer Photos

• LeBeouf to star in 'Indiana Jones 4'

Users review the new thriller 'Disturbia'

Scientists: Scrap the Internet and start over

Hi, Steven Sign Out

Mail Messenger Radio

Weather 81°F Local Horoscopes

Inspect Edit | swf1() < div#ad.ad < div.colpadding < div#right < div#rightcx < div#colcx < div#page < body.ywide < html

Console HTML CSS Script DOM Net YSlow Coder Options

```
<div class="colpadding">
  <div id="pa" class="md">
  <script type="text/javascript">
  <div id="ad" class="ad">
    <script language="javascript">
    <script src="http://us.js2.yimg.com/us.yimg.com/a/1-/java/pror
    <script language="javascript">
      <embed width="350" height="100" wmode="opaque" pluginspage="h
    <div id="lnkdiv" style="margin-top: 2px;">
    <script language="javascript">
      <noscript><img width=1 height=1 alt=""
      src="http://us.bc.yahoo.com/b?P=TPnA89j8eiGMUuvqRiBoEDxo2JExD(
  </div>
  <script type="text/javascript">
```

Style Layout DOM Options

```
body * {                                   www.yahoo.com (line 174)
    line-height: 1.22em;
}
```

Inherited from div#ad.ad

```
.ad {                                      www.yahoo.com (line 2125)
    text-align: center;
}

body * {                                   www.yahoo.com (line 174)
    line-height: 1.22em;
}
```

Inherited from div.colpadding

```
body * {                                   www.yahoo.com (line 174)
```

Done 6.820s YSlow

YSlow

performance lint tool

grades web pages for each rule

Firefox extension

Yahoo! internal tool

Yahoo! - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://www.yahoo.com/

Make Y! your home page

Yahoo! Voice: Make international calls for as low as 1¢/min.

at&t YAHOO!®

HIGH SPEED INTERNET

My Yahoo! My Mail

Web | Images | Video | Local | Shopping | more

Search: Web Search

Y! Answers: Ask a question | Answer questions

Page Options

- Autos
- Calendar
- Finance
- Games
- GeoCities
- Groups
- HotJobs
- Maps
- Movies
- Music

Featured | Entertainment | Sports | Life

Apr 14, 2007

Disturbing thoughts

Find out what Yahoo! users think of "Disturbia" – a modern twist on the classic "Rear Window." » Reviews

• Showtimes Trailer Photos

• LeBeouf to star in 'Indiana Jones 4'

Users review the new thriller 'Disturbia'

Scientists: Scrap the Internet and start over

Hi, Steven Sign Out

Mail Messenger Radio

Weather 81°F Local Horoscopes

Inspect | Performance | Stats | Components | Tools

Console HTML CSS Script DOM Net YSlow Coder Options

Performance Grade: A (94)

Expand All Collapse All

B	1. Minimize HTTP requests This page has 4 external JavaScript files.
A	2. Use edge computing
A	3. Add an Expires header
B	4. Gzip components These components are not gzipped: (1.9K) http://l.yimg.com/us.js.yimg.com/lib/bc/bc_2.0.3.js
B	5. Move CSS to the top

Done 5.185s A 179.9K

Conclusion

Takeaways

focus on the front-end

harvest the low-hanging fruit

you do control user response times

LOFNO – be an advocate for your users

Links

book: http://www.oreilly.com/catalog/9780596514211/
examples: http://stevesouders.com/examples/
image maps: http://www.w3.org/TR/html401/struct/objects.html#h-13.6
CSS sprites: http://alistapart.com/articles/sprites
inline images: http://tools.ietf.org/html/rfc2397
jsmin: http://crockford.com/javascript/jsmin
dojo compressor: http://dojotoolkit.org/docs/shrinksafe
HTTP status codes: http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html
IBM Page Detailer: http://alphaworks.ibm.com/tech/pagedetailer
Fasterfox: http://fasterfox.mozdev.org/
LiveHTTPHeaders: http://livehttpheaders.mozdev.org/
Firebug: http://getfirebug.com/
YUIBlog: http://yuiblog.com/blog/2006/11/28/performance-research-part-1/
http://yuiblog.com/blog/2007/01/04/performance-research-part-2/
http://yuiblog.com/blog/2007/03/01/performance-research-part-3/
http://yuiblog.com/blog/2007/04/11/performance-research-part-4/
YDN: http://developer.yahoo.net/blog/archives/2007/03/high_performanc.html
http://developer.yahoo.net/blog/archives/2007/04/rule_1_make_few.html