Globant

we are ready

Front End Engineer

Advanced HTML

Topics

Accessibility
WAI-ARIA
Progressive Enhancement
Semantic HTML
High Performance HTML & CSS

Accessible JavaScript Techniques

Now, you can see:

01 Designing for accessibility

http://youtu.be/r4voImh8Ev4

Building Accessibility (A11y)

Universal design

Design for all, everyone benefits

Accessibility is so commonplace, you don't notice

Curb cuts, access ramps

Closed captioning

Goal: accessibility becomes ubiquitous

IS A11y Starts with the Basics

Old school accessibility:

- Semantic markup
- Equivalent alternatives
  - Images
- Skip navigation
- Natural tab order

Semantic HTML is the Foundation

Conveying meaning of markup to assistive technology

- *Bad:* Page Title
- *Good:* <h1>Page Title</h1>

Key component of SEO

But wait…

Progressive Enhancement

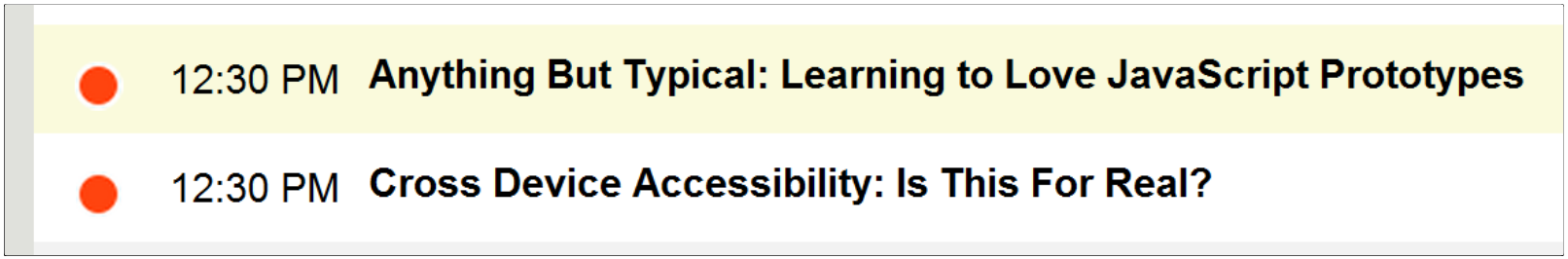
An essential part of making JS accessible is *not* to rely on it at all

All page content should be accessible without JS

Role of JS is purely to enhance a fully functioning and accessible site

Progressive Enhancement

Example



Problem:

Does not work w/o JS

Content remains hidden w/o JS

Progressive Enhancement

Example Under the hood:

```
<div id="event_row_406" class="event_odd">
    <img class="track_icon" src="/images/site-specific
    /interactive_track.png?1259927243" onclick="$('event_details_406').toggle()" alt="Interacti
    <div class="event_start_time" valign="top">
    <div id="event_name_406" class="event_name" onclick="$('event_details_406').toggle()">
    Anything But Typical: Learning to Love JavaScript Prototypes </div>
    <div class="event_venue_name"> Ballroom E </div>
</div>
<div></div>
```

Solutions:

- Establish basic page flow
- Change DIV's to meaningful elements
- Add JS events unobtrusively
  - `$('.eventName').click(function () {…});`

Progressive Enhancement

JS vs. Non JS CSS

Control the display of your widgets or hidden content using a simple script:

```
<body class="js-disabled">
<script type="text/javascript">
document.body.className =
     document.body.className.replace('js-disabled', 'js-
enabled');
</script>
```

CSS:

```
js-enabled .hidden {display:none;}
```

Start with a fully accessible site that works without JS

Progressive Enhancement

Control display of hidden content with JS script

Use unobtrusive JS techniques to keep layers separate and code maintainable

Now, you can see:

02 Progressive enhancement

http://youtu.be/d8qM7AiQUug

Now, you can see:

03 Progressive Enhancement 2 0 Nicholas Zakas

http://youtu.be/6Wr6dkkVIE0

Accessible JS

Key Points

Device independent navigation

Provide means to navigate a page and widgets using keyboard and mouse.

Access to pertinent page content

Hidden content(e.g. tooltips, hidden lists, menus, tabs) should be available to all users

User control or awareness

Give the user means to control their experience

Provide mechanisms for alerting the user to pertinent changes in the page.

Default browser behavior

Altering or disabling the normal functionality of the browser can confuse or disorient the user.

Keyboard Support Example

Goal: mimic default browser behavior

Keyboard navigation

Up/down arrows

Widget keyboard navigation recommendations

Focus Management

Roaming tabindex

Keyboard Support Example

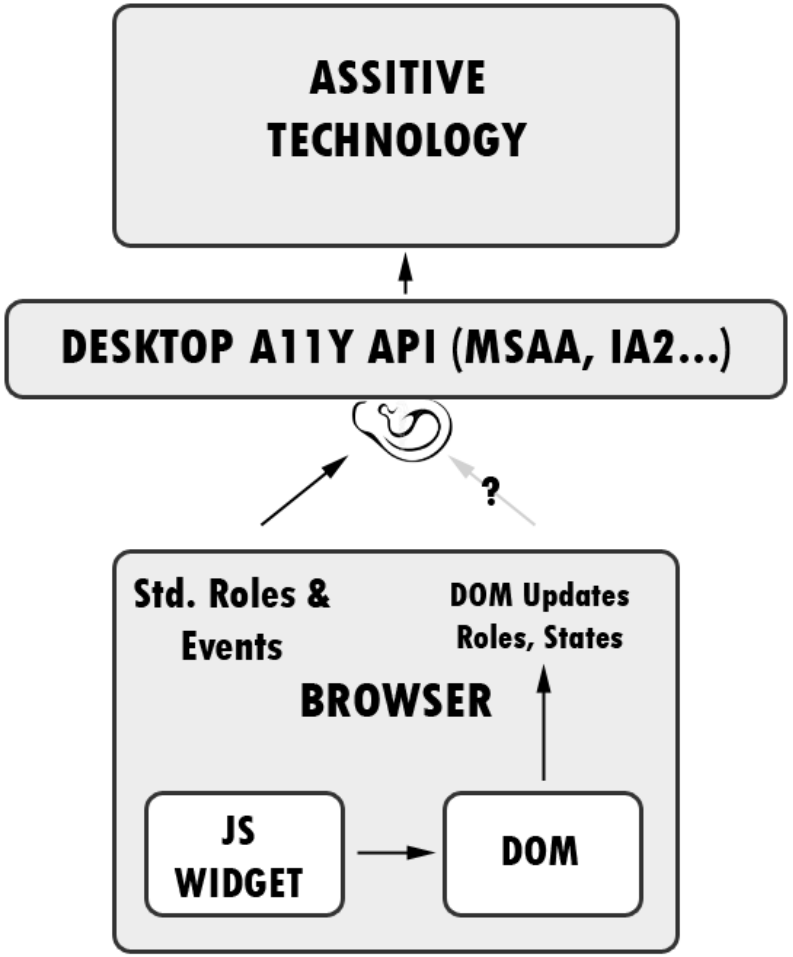
Keydown event listener code

```
me.Input.keydown(function (e) {
    me.Action = null;
    switch(e.keyCode) {
        case 38://up
        case 39://left
            me.Action = me.Actions.INCREMENT;
            break;
        case 37://right
        case 40://down
            me.Action = me.Actions.DECREMENT;
            break;
        case 35://end
            me.Action = me.Actions.MIN;
            break;
        case 36://home
            me.Action = me.Actions.MAX;
            break;
        case 33://page up
            me.Action = me.Actions.INCREMENT10;
            break;
        case 34://page down
            me.Action = me.Actions.DECREMENT10;
            break;
        default:
            return;
    }
    me.ChangeValue();
});
```

But that only gets you so far….

There's a gap…

There's no way to communicate common Web 2.0 events, concepts, states to AT



Communication Breakdown

HTML is limited

not intended for creation of applications, complex widgets

No built-in way to convey:

Widget roles

DOM updates(Ajax, display changes)

Bridging the Gap: WAI-ARIA

Addresses Web 2.0 Accessibility Issues
Backwards compatible, forward thinking
Developed at IBM, donated to W3C
At last call draft status
Support:
Firefox, IE 8
Initial implementation in Opera 9.5, Safari 4
Window-Eyes, JAWS and NVDA screen readers
Used by IBM, Dojo, Yahoo, Google

ARIA Details

Roles
States and Properties
Keyboard Support
Landmarks
Live Regions

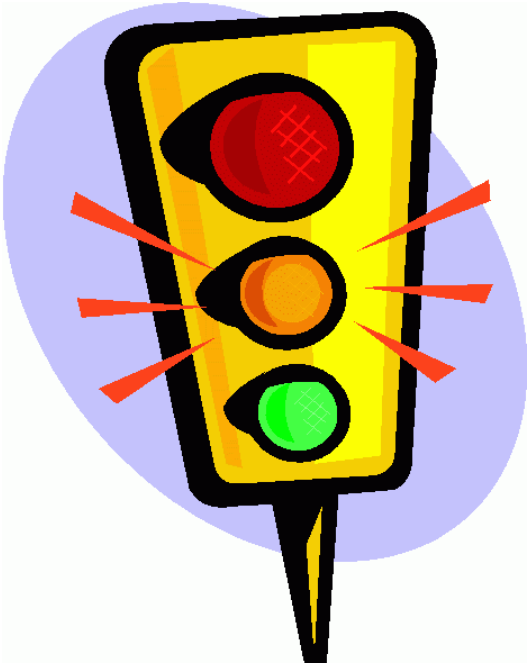
ARIA Roles

Add semantics to scripted user interface (UI) elements

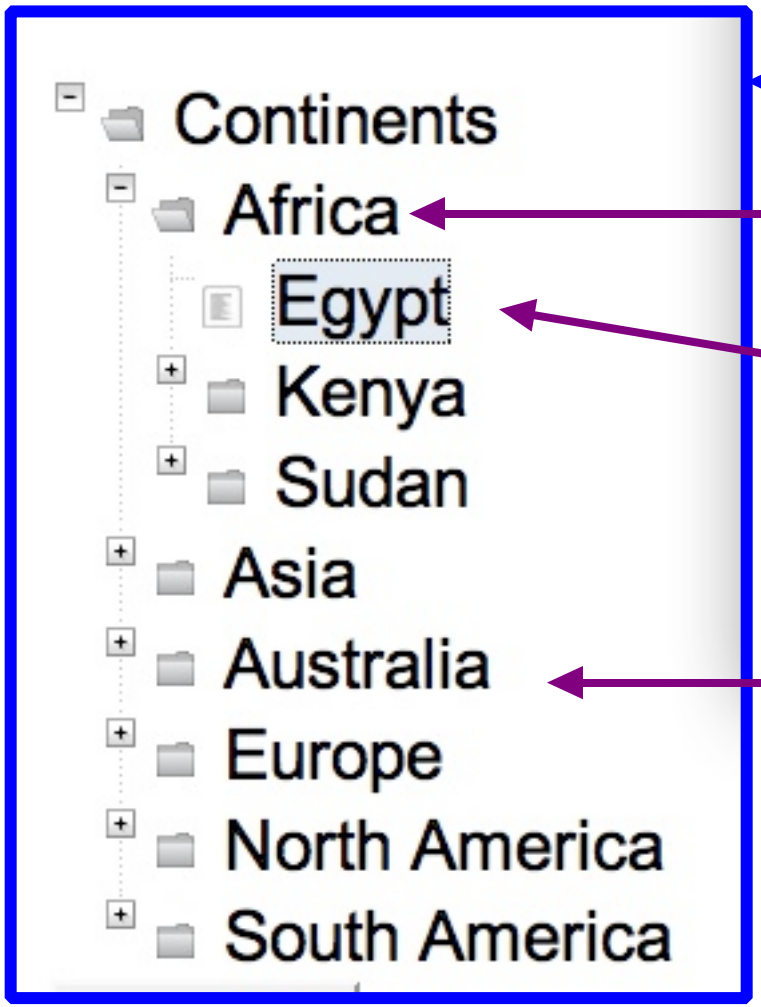
A passport or caller Id for your controls!

ARIA States and Properties

Additional Details about the control
Label, multi-selectable, readonly, required, etc.
States that may change over time
Expanded/collapsed, checked, hidden, etc.



ARIA Example Tree Structure



Role = tree
(on outer container)

Role = treeitem
aria-expanded=true
(on open Africa node)

Role = treeitem
aria-selected=true
(on highlighted Egypt child node with no children)

Role = treeitem
aria-expanded=false
(on closed Australia node)

ARIA – Keyboard Support

Make items focusable via tabindex attribute
Add keyboard event handling

- Mimic the behavior of desktop controls
- Minimize tab key navigation

ARIA – Live Regions

Identify and announce updated regions to support Ajax

Announce new and important information and additions

ARIA – Live Regions

Perceivable sections are identified with region role
Live indicates region is updated
Values of: Off, Polite, Assertive
Atomic identifies the extent of updates
True – entire region is updated and relevant
False – only changed element needs to be presented to user
Controls identifies the element which triggers an update

ARIA Example – Live Regions

Editing auto save notification

live=polite; atomic=true;

Server maintenance notification

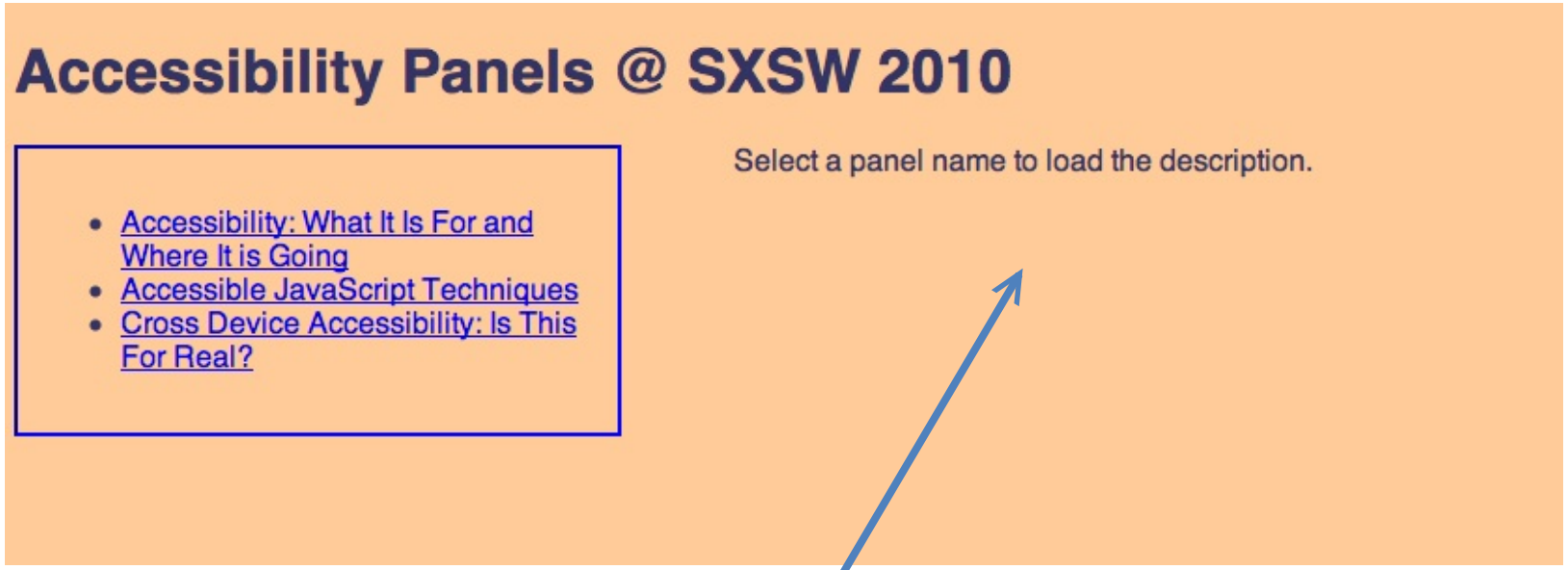
Live=assertive; atomic=true;

New mail - speak new entries

Live=polite; atomic=false;

Wile E. Coyote's Gravity Lessons

ARIA – Live Regions




```
<div id="main" role="main">
        <div id="content" role="document" aria-live="assertive">
                Select a panel name to load the description.
        </div>
</div>
```

ARIA – Landmarks

Identify important regions on the page

ARIA – Landmark Roles

Makes finding and navigating to sections of the page easier

Application

Banner

Complementary

Contentinfo

Main

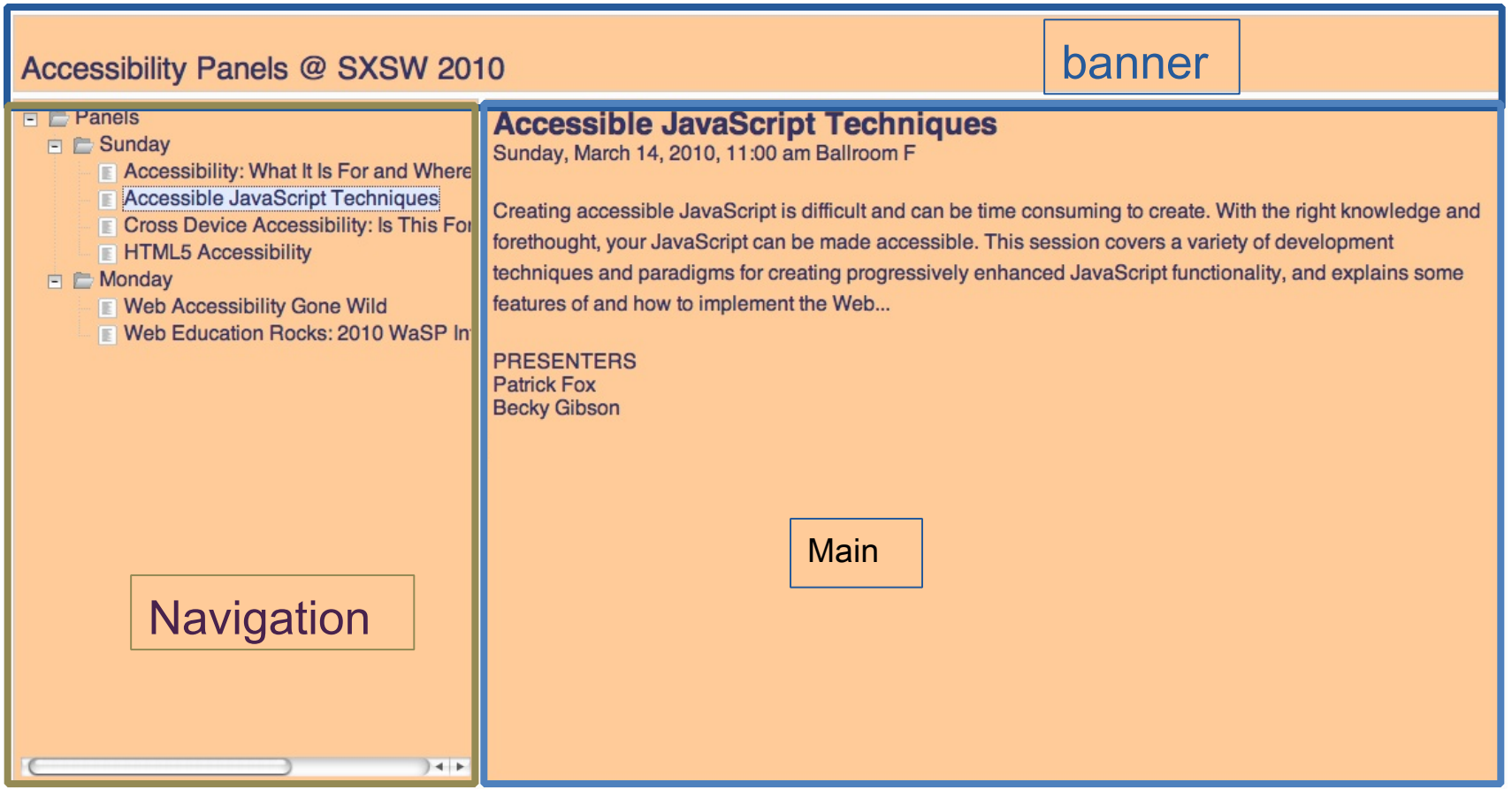
Navigation

Search

Header

Example: `<div role="navigation">`

ARIA – Landmark Example



Now, you can see:

04 Accessibility Post Web 2 0 Michael Cooper

http://youtu.be/dpMyLCAvT_I

All of this sounds like a lot of work….

...and it is more work, but not a lot.

Use a JavaScript Toolkit!

Dojo
JQuery
YUI
GWT

They make progressive enhancement, unobtrusive JS and accessible JS **EASY(ER)** to implement

Dojo – What is it?

Open Source JavaScript Toolkit
"Easy" Ajax
Data Binding
Full event system
Browser abstraction layer
User Interface Widgets
Liberally Licensed
Asynchronous Loading

A11y Support in Dojo Core Widgets

Keyboard support

Screen reader support

Low Vision Support

High Performance Web Sites

14 rules for faster-loading pages

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

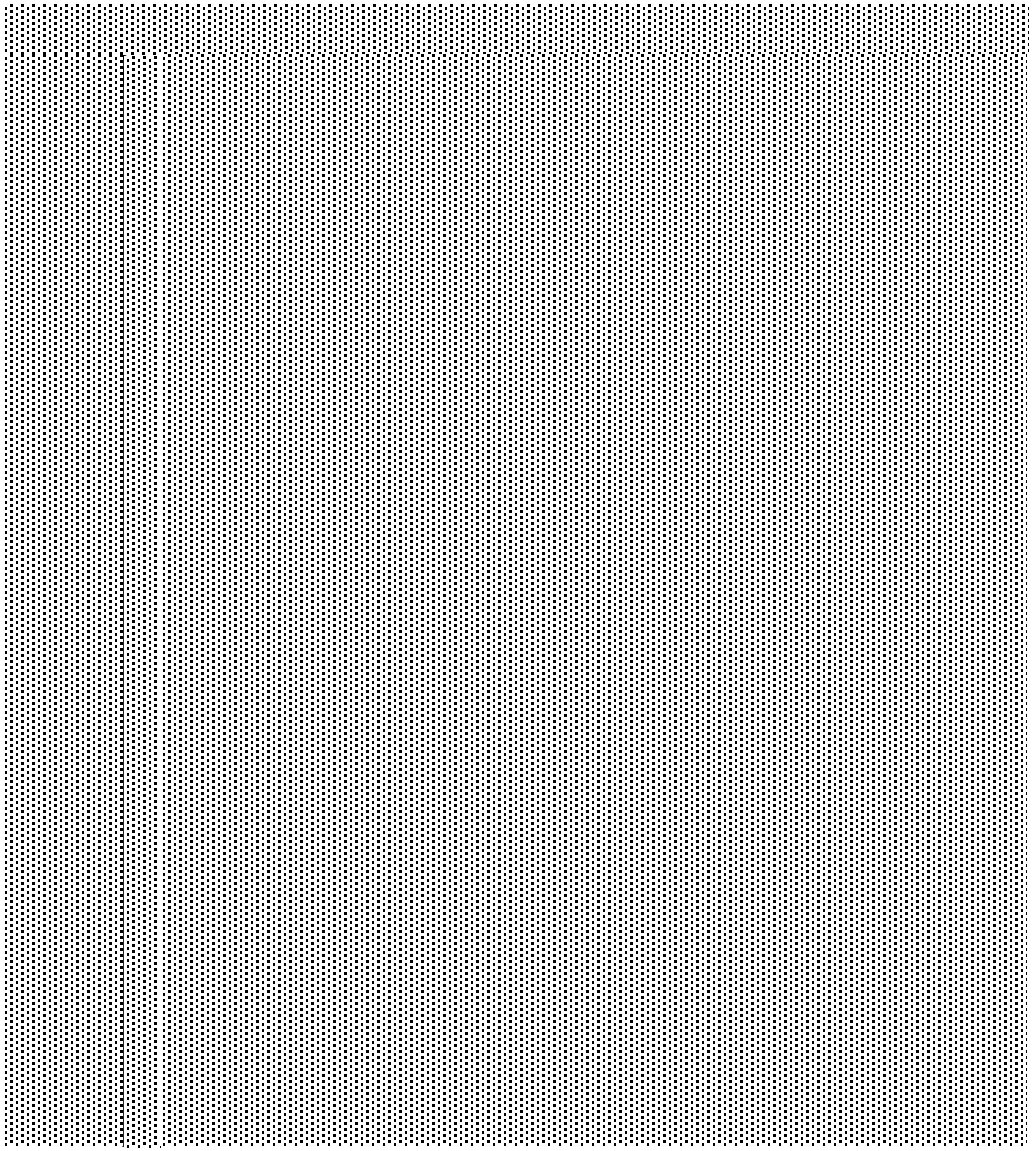
response time

efficiency

Current focus is response time

Of web products

The Importance of Front-End Performance



Back-end vs. Front-end

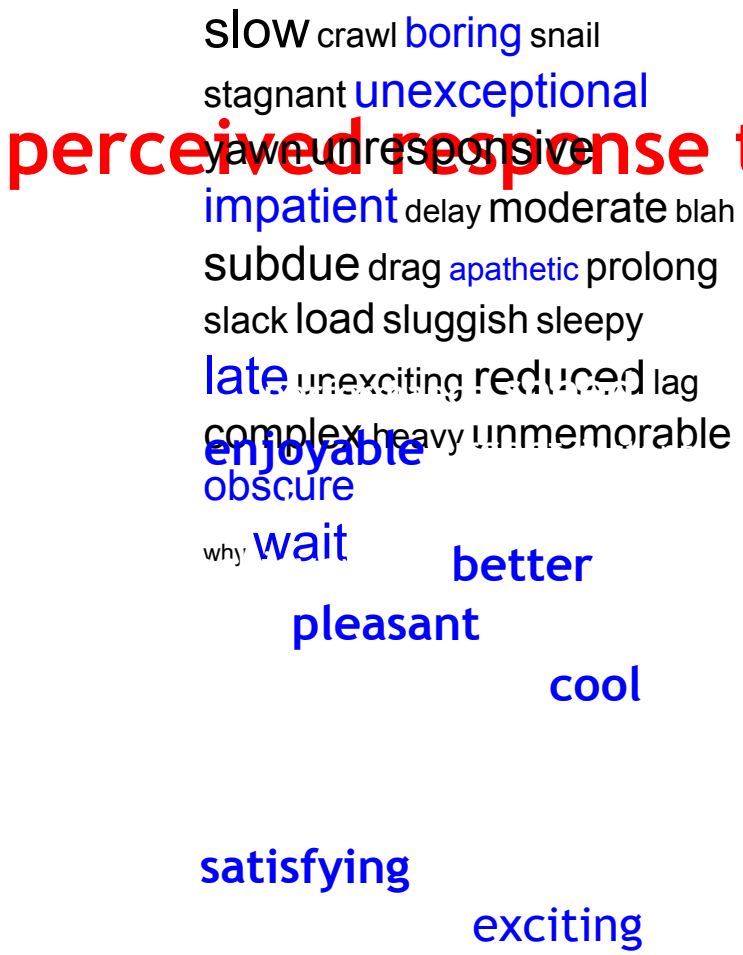
	Empty Cache	Full Cache
amazon.com	82%	86%
aol.com	94%	86%
cnn.com	81%	92%
ebay.com	98%	92%
google.com	86%	64%
msn.com	97%	95%
myspace.com	96%	86%
wikipedia.org	80%	88%
yahoo.com	95%	88%
youtube.com	97%	95%

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



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.

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

1

user requests
www.yahoo.com

2

user requests
other web pages

3

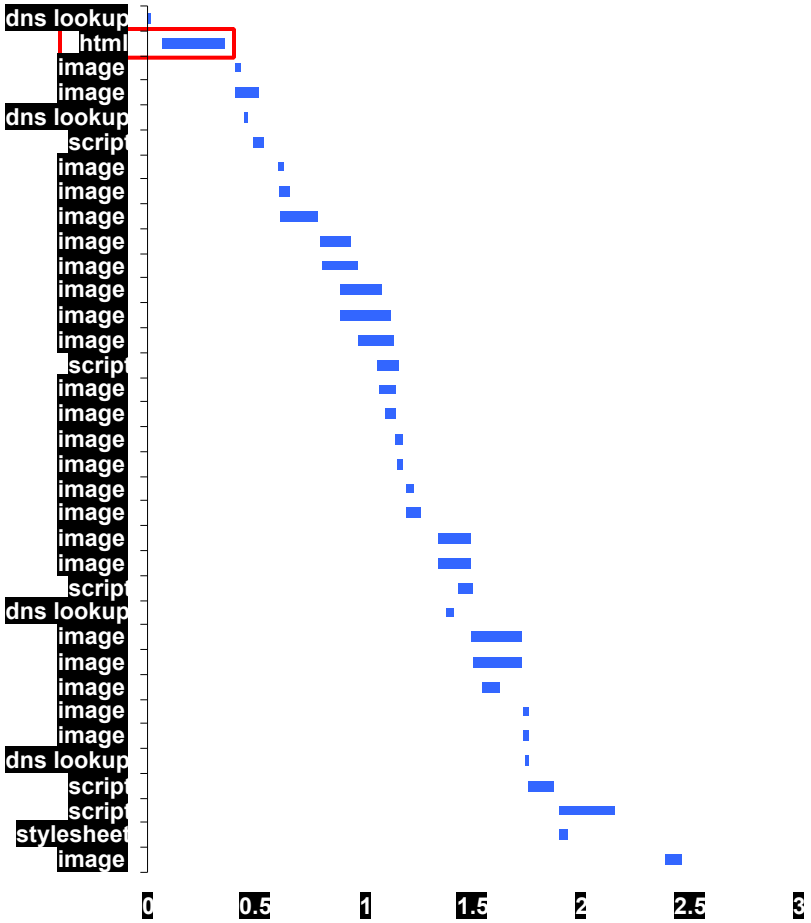
user re-requests
www.yahoo.com

Empty vs. Full Cache

1

user requests
www.yahoo.com

with an empty cache

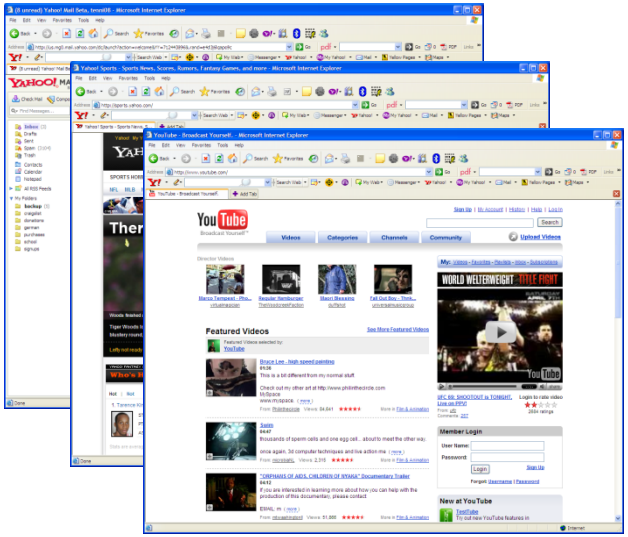


Empty vs. Full Cache

1 **2** **3**

user requests www.yahoo.com

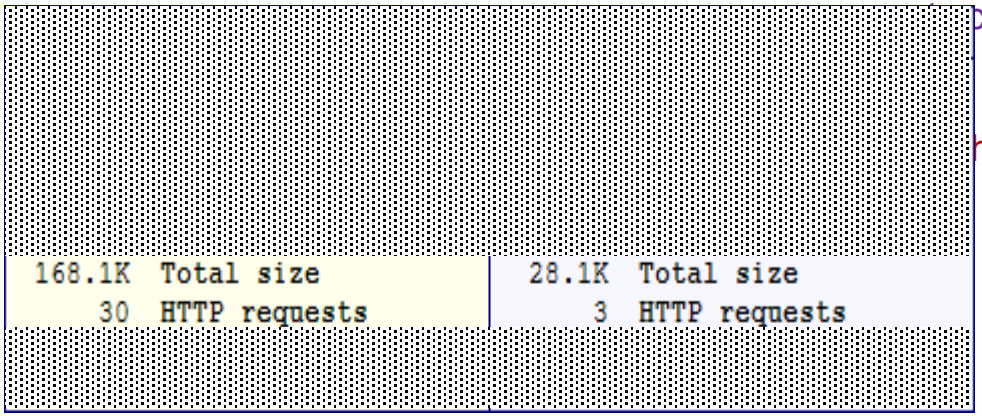
user requests other web pages



user re-requests www.yahoo.com

Empty vs. Full Cache

Empty vs. Full Cache



cache
2.4 seconds

he
0.9 seconds

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*?

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
```

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?

unique users with at least one 200 response
total # unique users

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

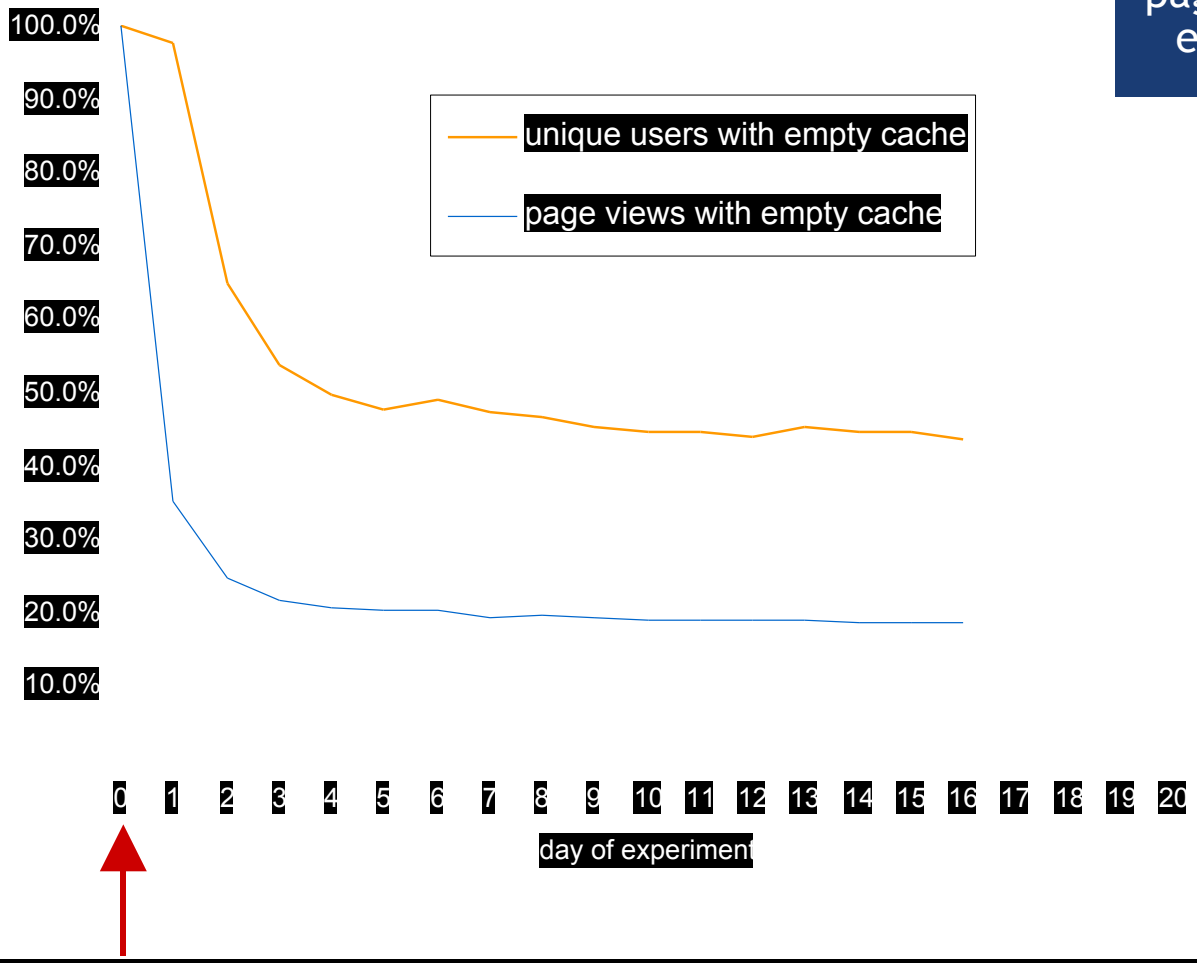
total # of 200 responses
# of 200 + # of 304 responses

1 px

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.

HTTP Quick Review

1

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
```

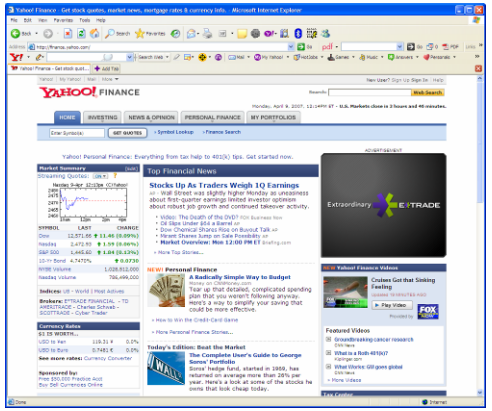
HTTP Quick Review

1

2

user requests
www.yahoo.com

user requests
finance.yahoo.com



HTTP request header sent by the browser:

```
GET / HTTP/1.1
Host: finance.yahoo.com
Cookie: C=abcdefghijklmnopqrstuvwxyz;
```

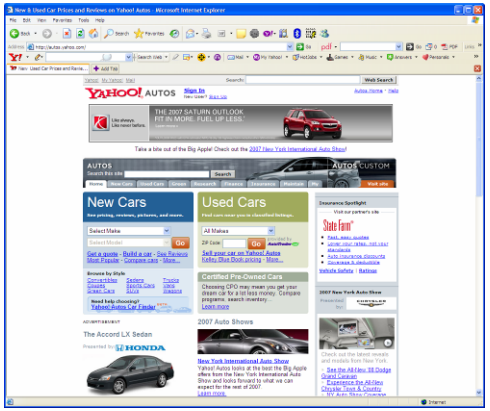
HTTP Quick Review

1

3

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
Cookie: C=abcdefghijklmnopqrstuvwxyz;
```

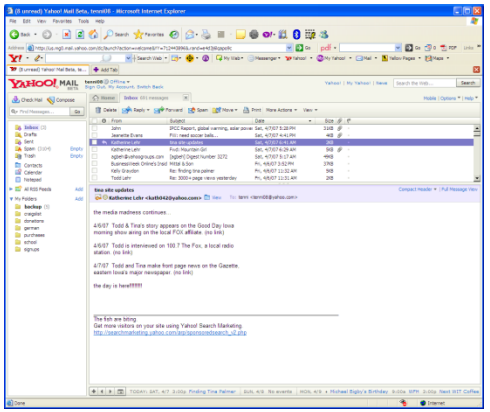
HTTP Quick Review

1

4

user requests
www.yahoo.com

user requests
mail.yahoo.com



HTTP request header sent by the browser:

```
GET / HTTP/1.1
Host: mail.yahoo.com
Cookie: C=abcdefghijklmnopqrstuvwxyz;
```

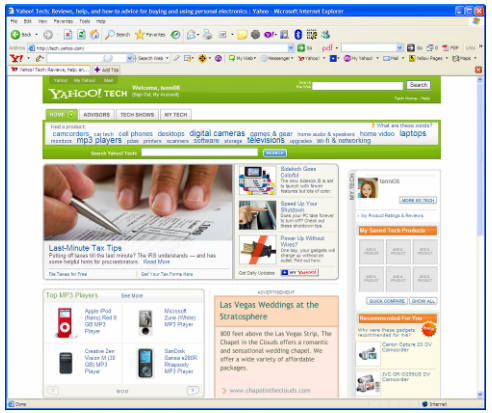
HTTP Quick Review

1

5

user requests
www.yahoo.com

user requests
tech.yahoo.com



HTTP request header sent by the browser:

```
GET / HTTP/1.1
Host: tech.yahoo.com
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

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

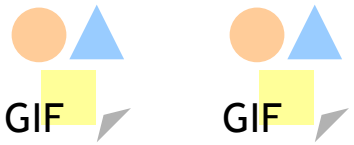
Set cookies at appropriate domain level

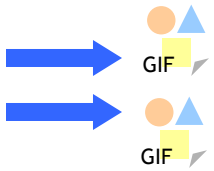
Set Expires date appropriately

- Earlier date or none removes cookie sooner

Parallel Downloads

Two components **in parallel** **per hostname**



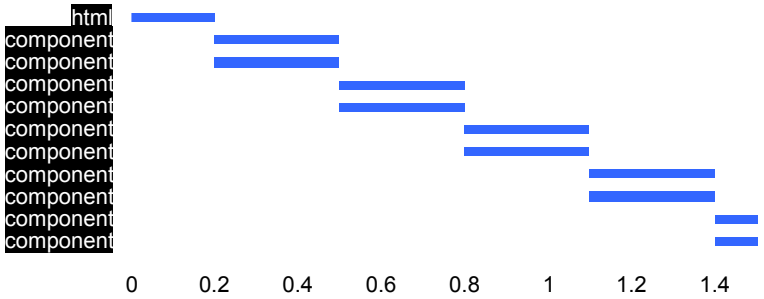


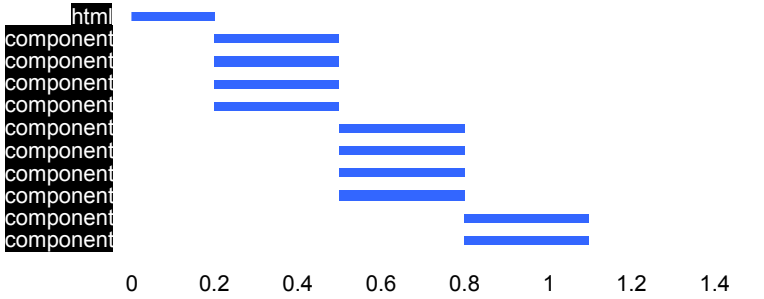
Parallel Downloads

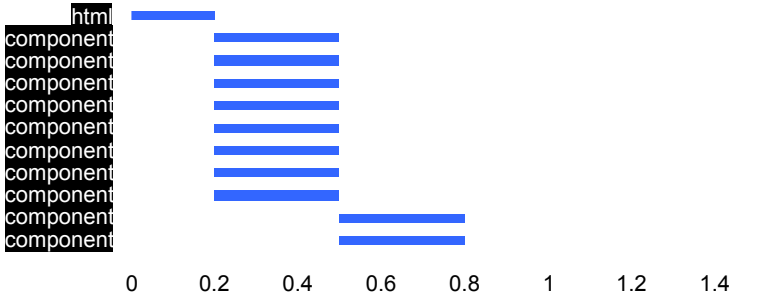
Two in parallel

Four in parallel

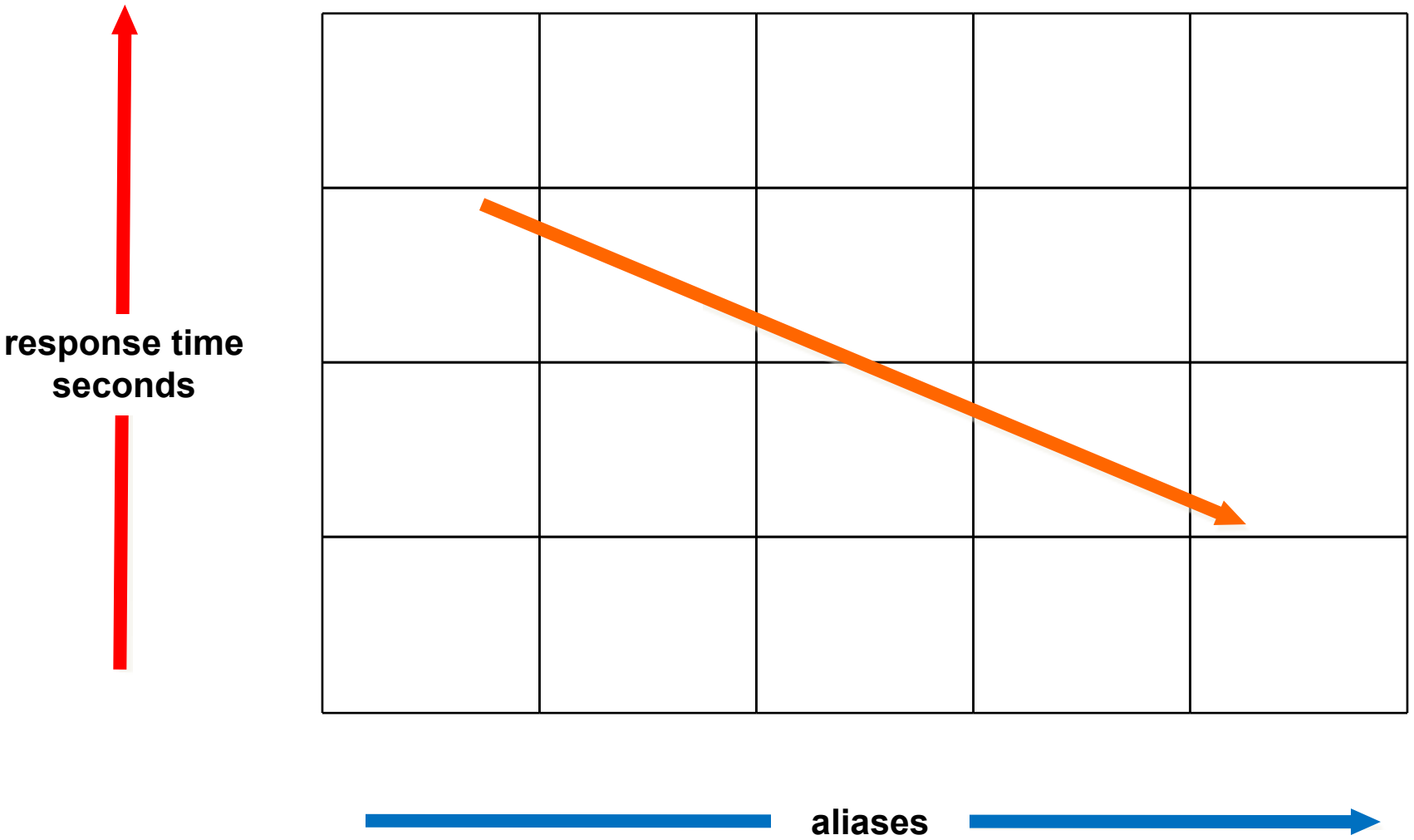
Eight in parallel







Maximizing Parallel Downloads



Experiment Takeaways

Consider the effects of CPU thrashing

DNS lookup times vary across ISPs and geographic locations

Domain names may not be cached

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

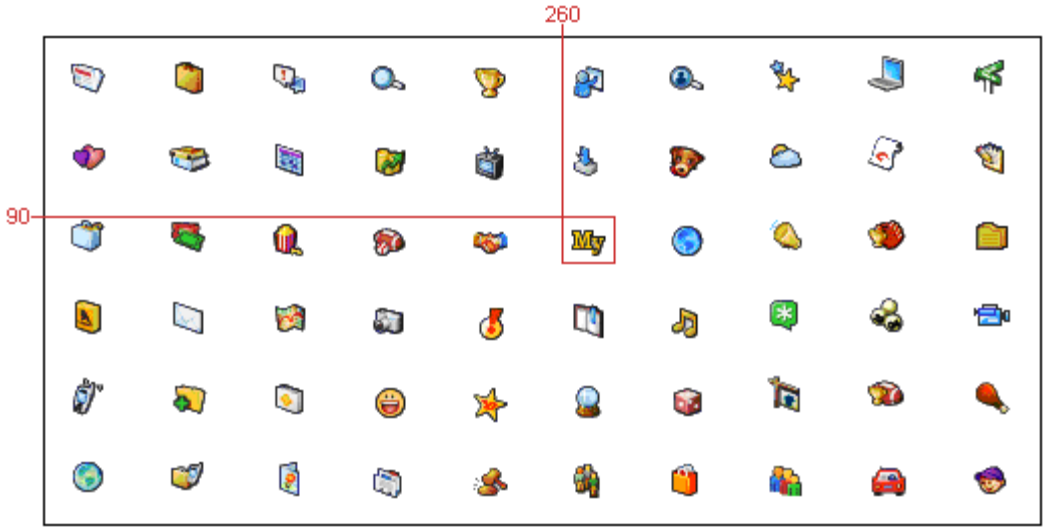
```
<img usemap="#map1" border=0 src="/images/imagemap.gif">
<map name="map1">
  <area shape="rect" coords="0,0,31,31" href="home.html" title="Home">
   …
</map>
```

drawbacks:

must be contiguous

defining area coordinates – tedious, errors

CSS Sprites -Preferred




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

size of combined image is less

Inline Images

`data:` URL scheme

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

```
<IMG ALT=”Red Star”
SRC="
FvcPewsO9VVfajo+w6O/zl5estLv/8/
AAAAAAAAAAAAAAAACH5BAEAAAsALAAAAAAMAAwAAAQzcElZyryTEHyTUgknHd9xGV+qKsYirK
kwDYiKDBiatt2H1KBLQRFIJAIKywRgmhwAIlEEADs=">
```

avoid increasing size of HTML pages:

put inline images in cached stylesheets

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

A content delivery network (CDN) is a collection of web servers distributed across multiple locations to deliver content more efficiently to users.

The server selected for delivering content to a specific user is typically based on a measure of network proximity.

For example, the server with the fewest network hops or the server with the quickest response time is chosen.

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

You can affect users' download times
90%+ of browsers support compression

Rule 4: Gzip Components

The time it takes to transfer an HTTP request and response across the network can be significantly reduced by decisions made by front-end engineers.

It's true that the end-user's bandwidth speed, Internet service provider, proximity to peering exchange points, etc. are beyond the control of the development team.

But there are other variables that affect response times.

Compression reduces response times by reducing the size of the HTTP response.

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 Configurations

HTTP request

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

HTTP response

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

Rule 5: Put CSS at the top

stylesheets block rendering in IE

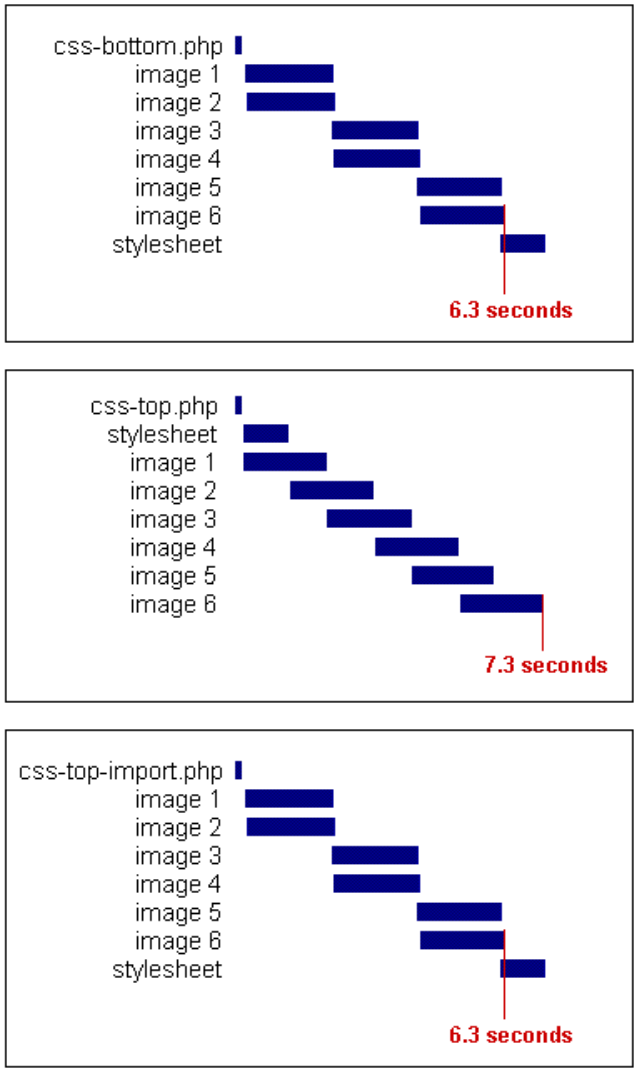
http://stevesouders.com/examples/css-bottom.php

solution: put stylesheets in HEAD (per spec)

avoids Flash of Unstyled Content

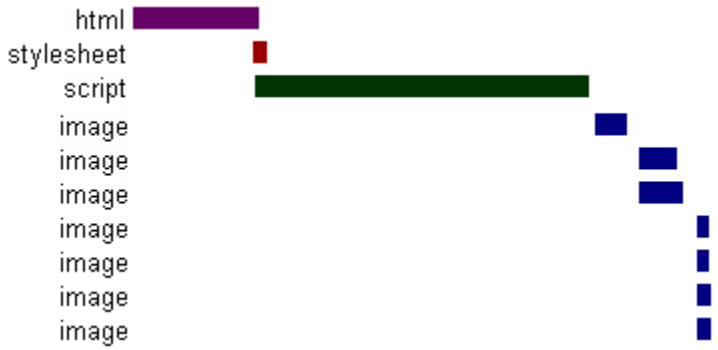
use LINK (not @import)

Slowest is fast



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

CSS expressions are a powerful (and dangerous) way to set CSS properties dynamically. They were supported in Internet Explorer starting with version 5, but were deprecated starting with IE8.

As an example, the background color could be set to alternate every hour using CSS expressions:

```
background-color: expression( (new Date()).getHours()%2 ? "#B8D4FF" : "#F08A00" );
```

As shown here, the expression method accepts a JavaScript expression. The CSS property is set to the result of evaluating the JavaScript expression.

Rule 7: Avoid CSS expressions

The expression method is ignored by other browsers, so it is useful for setting properties in Internet Explorer needed to create a consistent experience across browsers.

The problem with expressions is that they are evaluated more frequently than most people expect. Not only are they evaluated when the page is rendered and resized, but also when the page is scrolled and even when the user moves the mouse over the page.

Adding a counter to the CSS expression allows us to keep track of when and how often a CSS expression is evaluated. Moving the mouse around the page can easily generate more than 10,000 evaluations.

Rule 7: Avoid CSS expressions

One way to reduce the number of times your CSS expression is evaluated is to use one-time expressions, where the first time the expression is evaluated it sets the style property to an explicit value, which replaces the CSS expression.

If the style property must be set dynamically throughout the life of the page, using event handlers instead of CSS expressions is an alternative approach.

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

inline in front page

download external files after onload

Post-Onload Download

```
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

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

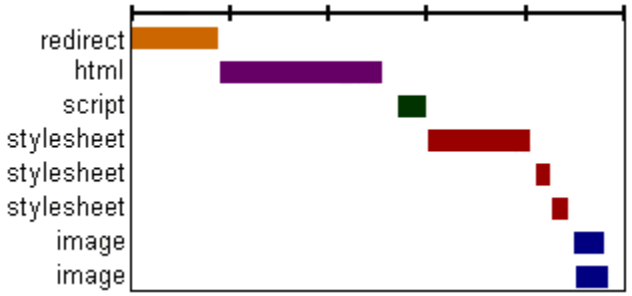
Rule 11: Avoid redirects

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



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

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

Rule 13: Turn off ETags

Unique identifier returned in response

HTTP/1.1 200 OK
Last-Modified: Tue, 12 Dec 2006 03:03:59 GMT
ETag: "10c24bc-4ab-457e1c1f"

Used in conditional GET requests

GET /i/yahoo.gif HTTP/1.1
If-Modified-Since: Tue, 12 Dec 2006 03:03:59 GMT
If-None-Match: "10c24bc-4ab-457e1c1f"
HTTP/1.1 304 Not Modified

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/

XHR, JSON, iframe, dynamic scripts can still be cached, minified, and gzipped

a personalized response should still be cacheable by that person

Rule 14: Make AJAX cacheable and small

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

Focus on the front-end

Harvest the low-hanging fruit

You do control user response times

LOFNO – be an advocate for your users

Thank you!

Globant
we are ready