Deep Dive into Browser Performance

Ilia Alshanetsky

@iliaa

Slides — http://joind.in/14232

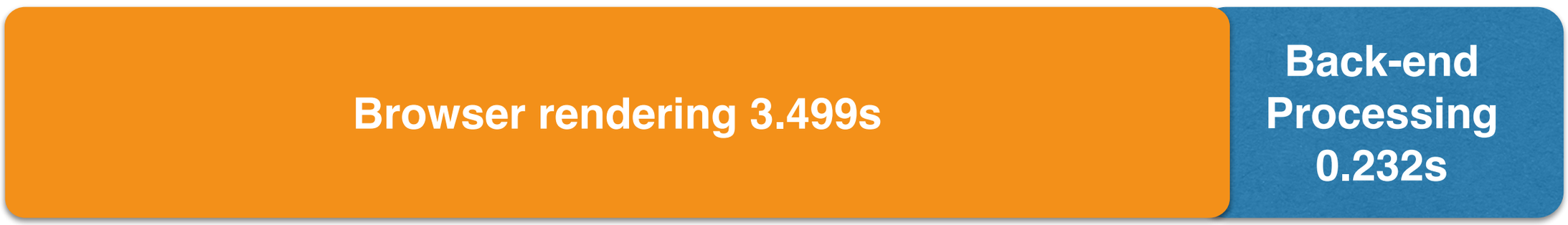
Me, Myself and I

PHP Core Developer

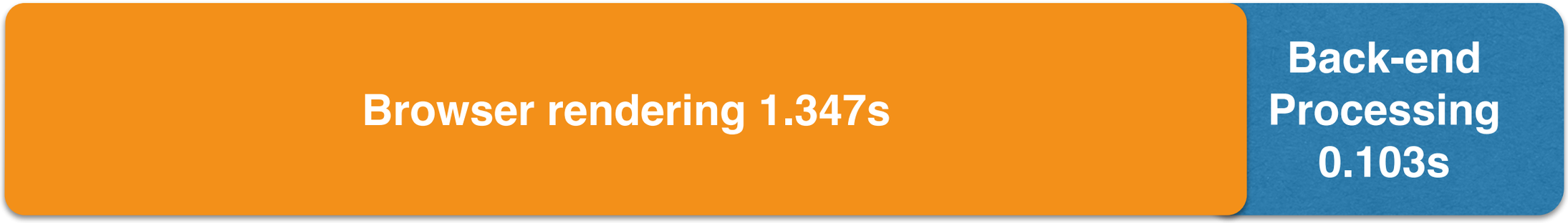
Author of
Guide to PHP Security

CIO at Centah Inc.

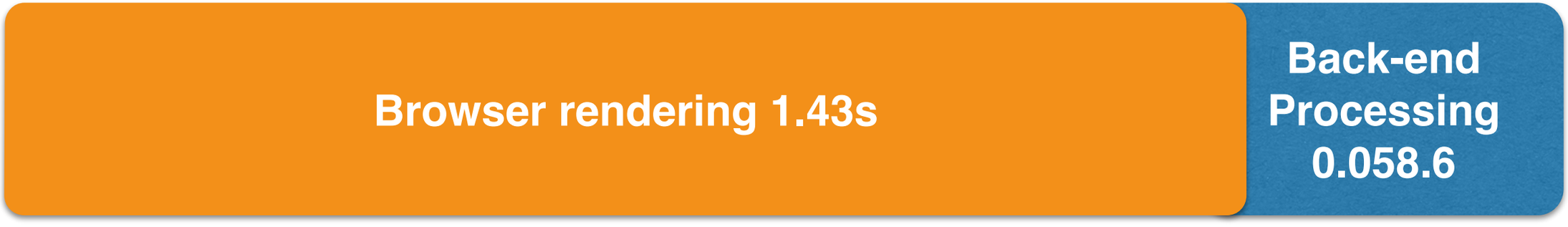
Why Browser Performance Matters?



phpconference.nl - Total Page Load - 3.73s



PHP.net - Total Page Load - 1.45s



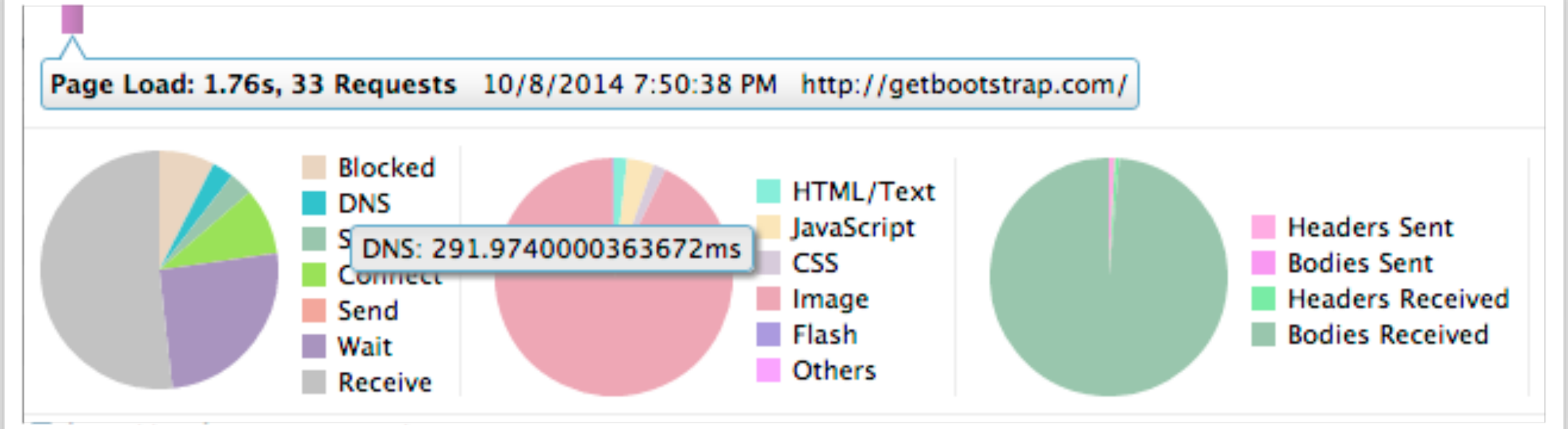
Github - Total Page Load - 1.43s

What Takes All This Time?

1. **DNS**
2. **HTTP + SSL Negotiation**
3. **JavaScript Processing**
4. **CSS Rendering**
5. **Image Rendering**
6. **DOM Rendering**

by James Balderson

DNS



DNS may take up-to 20% of 1st page load!

DNS Prefetching

Instruct the browser to pre-resolve (sub)domain from which you intend to retrieve resources.

```
<link rel="dns-prefetch" href="//mydomain.tld" />
```

Supported By:

Firefox 3.5+
Chrome
Safari 5+
IE 9+

DNS Based Optimizations

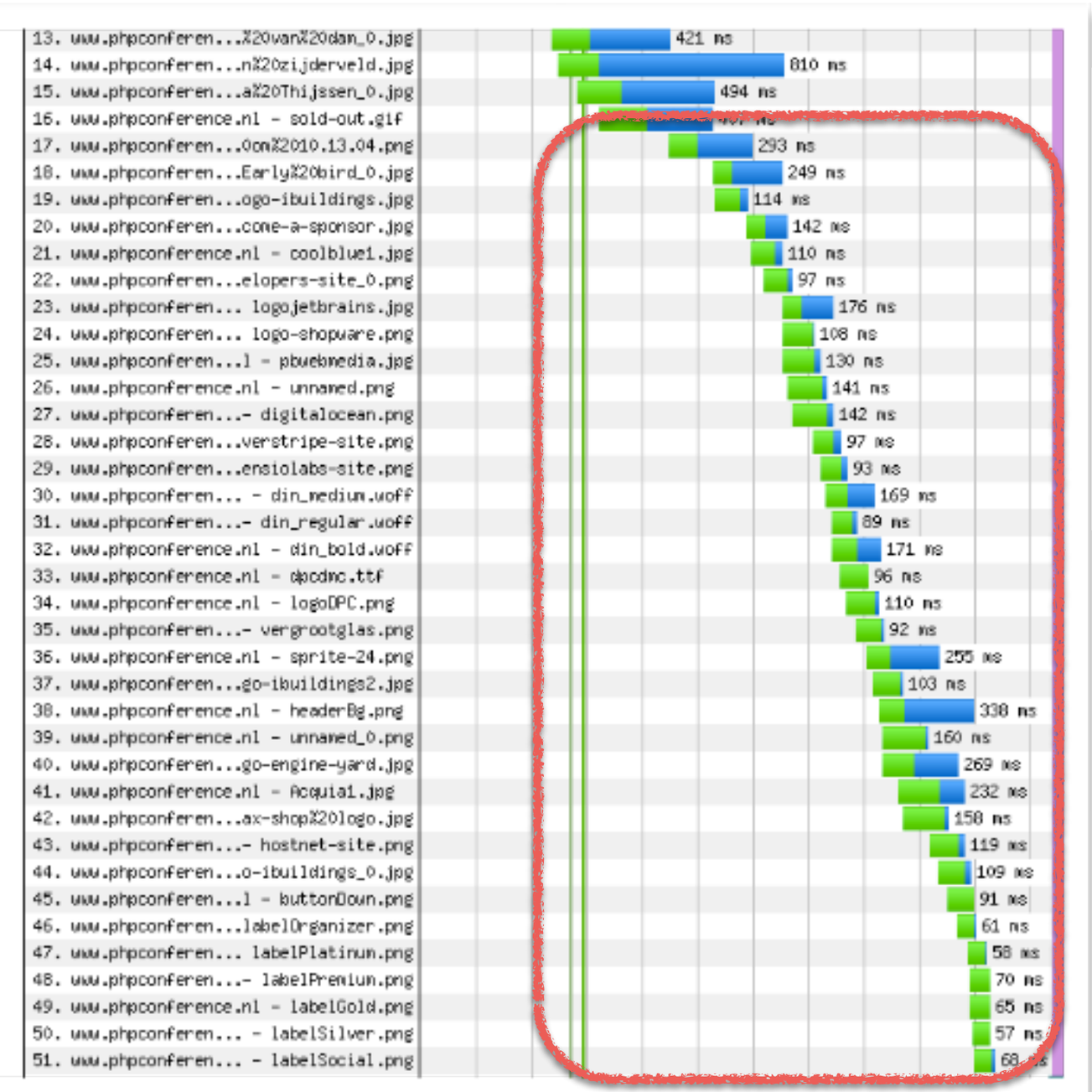
1. Use Embedded images via data:image

2. Defer loading of external resources

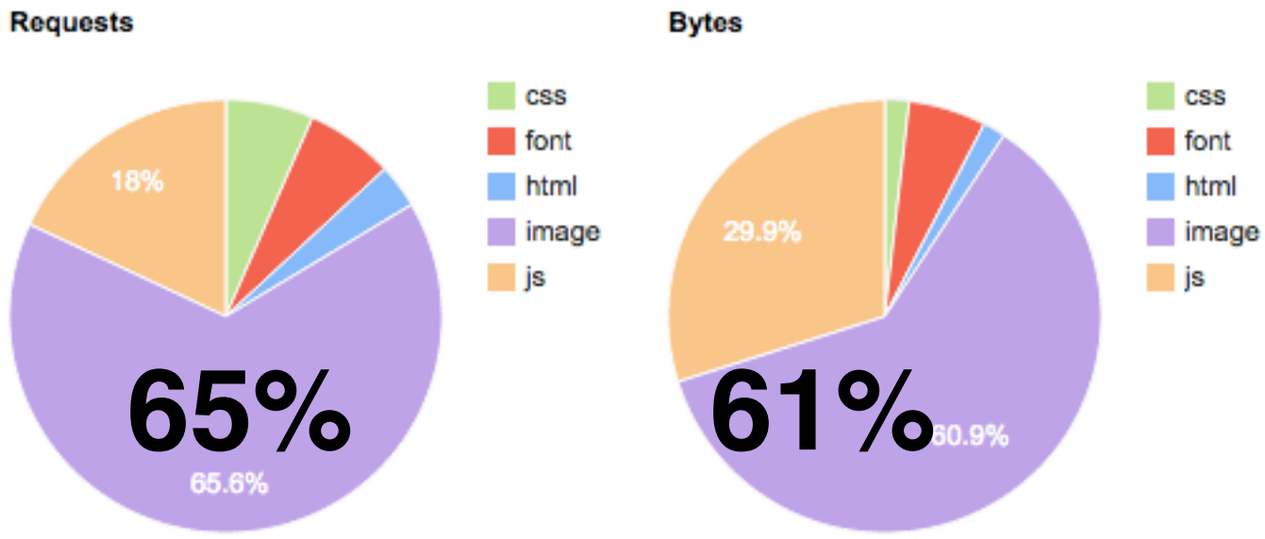
3. Avoid multi-domain CDNs *(7 for DPC)*

4. Sub-domains still need to be resolved

Use Sprites & Compress



phpconference.nl



40 images = 1 Request

64% compression savings (205kb reduction)

12.5% sprite savings (14kb reduction)

!

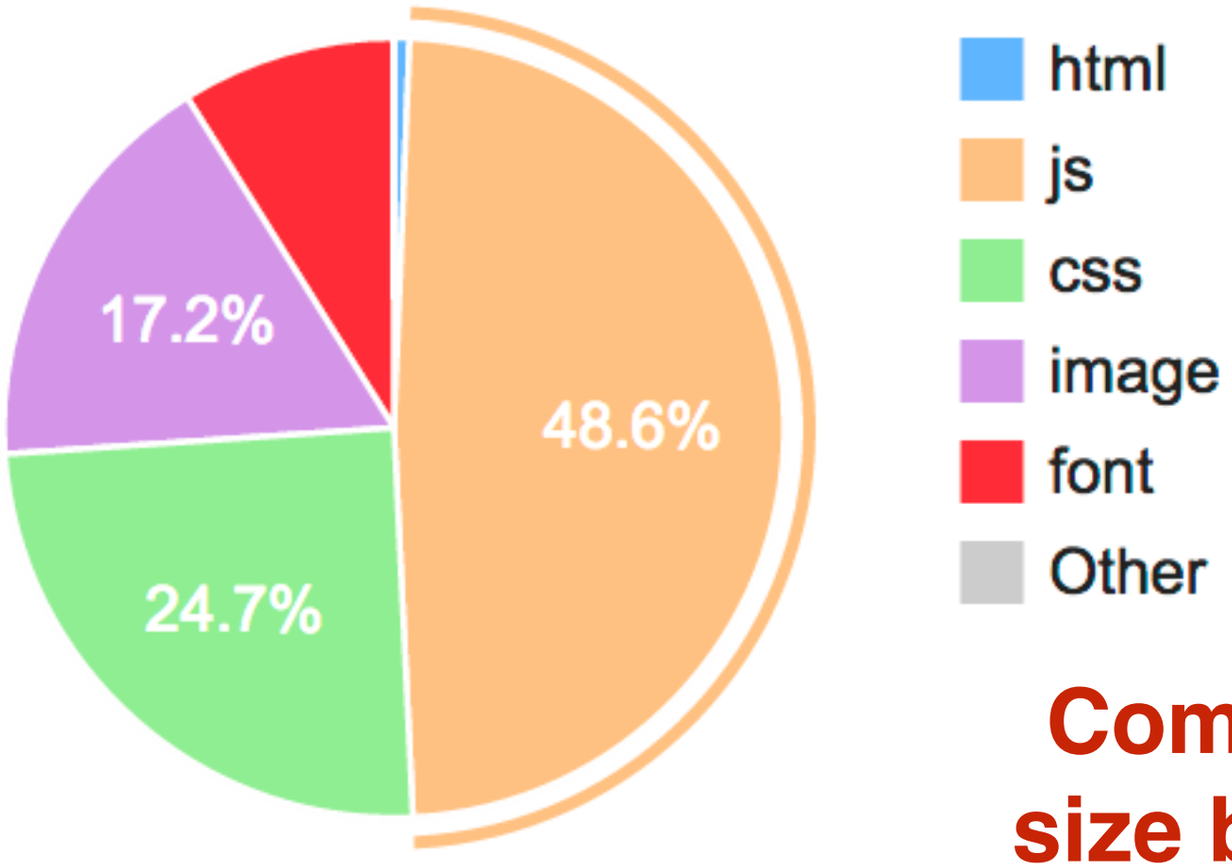
http://www.spritebox.net/

http://spritepad.wearekiss.com/

Profile Page Loading

- Use Your Browser
  - **Developer Tools or Equivalent**
- Do Remote Tests
  - **http://www.webpagetest.org/**
  - **https://developers.google.com/speed/pagespeed/**
  - **https://www.modern.ie/en-us**
- Actual User Profiling
  - **http://www.lognormal.com/boomerang/doc/**
  - **Use Web-Timing API directly**

Compression For The Win!



1,394 KB
59 requests,
4.63 seconds to load

Compression Reduces data-size by >50% and makes page load in 2.1 seconds!

Use gzip compression

965.8 KB total in compressible text, savings = 695.2 KB

Compress Images

171.6 KB total in images,savings = 51.8 KB

phpconference.nl via http://www.webpagetest.org

	Load Time	First Byte	DOM Elements	Document Complete			Fully Loaded		
				Time	Requests	Bytes In	Time	Requests	Bytes In
First	3.731s	0.232s	347	3.731s	63	992 KB	3.866s	63	1,061 KB
Repeat	3.480s	1.865s	347	3.480s	13	19 KB	3.480s	13	101 KB

Cache, Cache, Cache

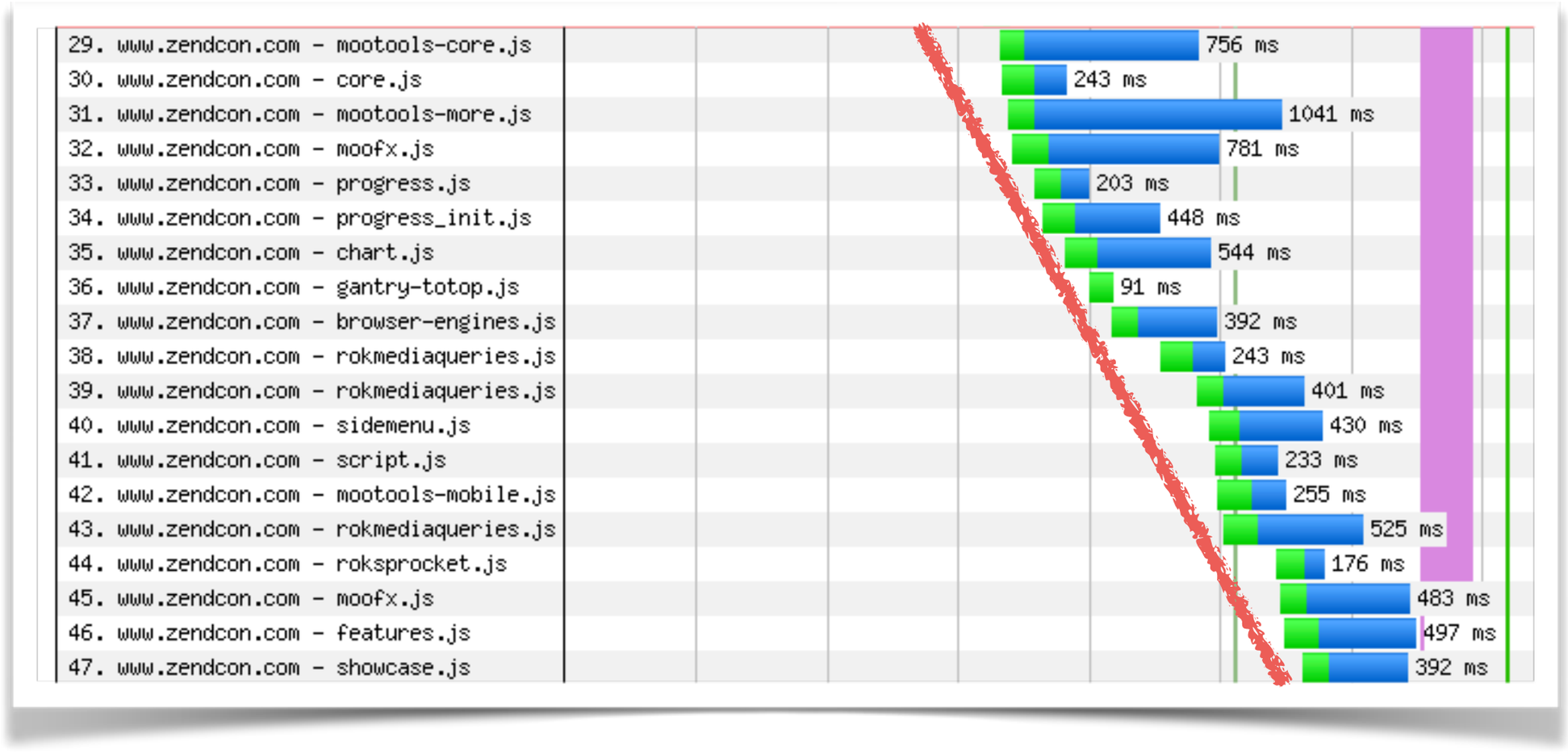
Set **max-age** or **expires** headers

Value should be at least **30 days**

To prevent stale content, **use unique file names** on new deployments for changed files.

Your goal is that 2nd page load only queries the server for the dynamic content!

JavaScript



JavaScript is loaded synchronously, so compact your files into a single compressed file!

JavaScript

Combination & minifying of JS files is best achieved with:

- Closure Compiler
  - http://goo.gl/8MVOIJ

- YUI Compressor
  - http://refresh-sf.com/yui/
  - http://yui.github.io/yuicompressor/

- PHP Based
  - https://github.com/tedious/JShrink

JavaScript

Don’t over-do combining of JS Files!

- Unnecessary data loading
- Decompression Overhead
- Extra JS Compilation

Micro-Case Study: SlashDot.org

One “BIG” JavaScript file

71kb compressed, 251kb actual size

199ms to receive

37ms to process

21.3% of total page load, 16% of total page size

< 10% of loaded JS code is executed

JavaScript

Only load up-front what you absolutely need

Defer loading of everything else via RequireJS

```
<head>
  <script src="scripts/require.js"></script>
</head>

require.config({
    baseUrl: 'js/lib',
    paths: { jquery: ‘jquery-1.11.1' }
});

define(['lib/jquery'], function ($) {...});
```

http://requirejs.org/

If you can’t win, cheat!


```
$(document).ready(function() {
  setTimeout(function() {
    $.get( “your-file.js” );
  }, 2000);
};
```

General JS Tips

1. **Avoid Xpath, reference/search by ID**

2. **Setup events pre-load as opposed to post-load**

```
onkeyup="js_function()" vs $("input").each(function() {});
```

3. **For Grids only load the data to be displayed**

4. **innerHTML is not always faster than DOM**

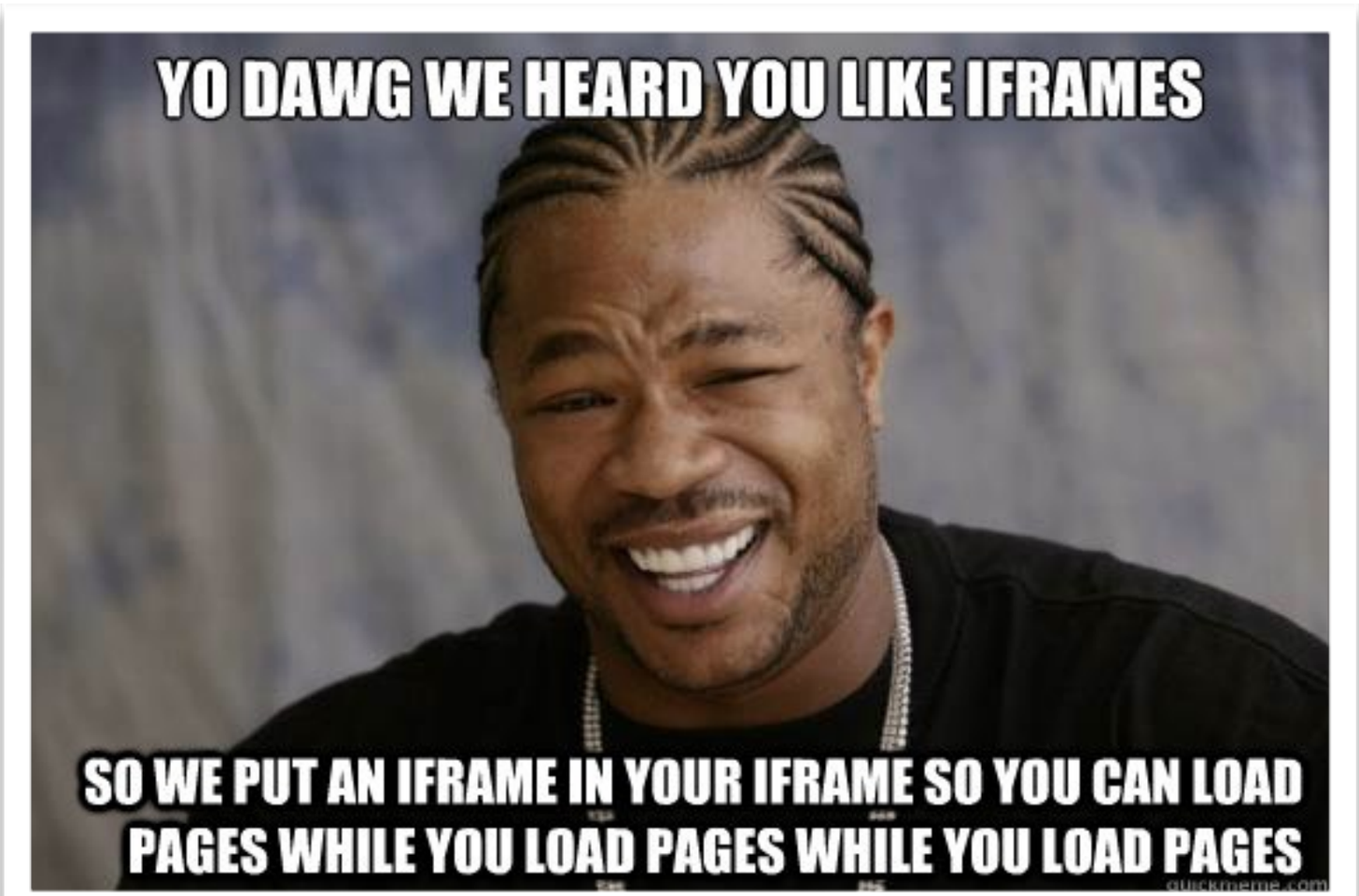
http://jsperf.com/dom-vs-innerhtml/37

General JS Tips

- Most browsers leak memory with JS, avoid the common culprits:
  - ✦ Use closures
  - ✦ Avoid passing objects (can result in circular references)
  - ✦ Avoid global variables

General JS Tips

- Help browser to make use of multiple CPUs by using iframes to embed complex components such as grids.



CSS

* **Minimize**
* **Combine**
* **Compress**

* **Don’t fear inlined (<style>) CSS**

Avoid Repaints & Reflows

- Changes to DOM nodes
- Hiding DOM nodes
- Actions that extend the page (causes scroll)
- Changes to colour, background and outline properties

Merge Style Changes

```
// slowest
el.style.left = "10px";
el.style.top  = "10px";

// getting better
el.className += " top-left-class";

// best
el.style.cssText += "; left: 10px; top: 10px;";
```

Peekaboo Trick

```
var me = $("#el");
me.hide();

// make various changes to DOM/Content

me.show();
```

Dolly Trick

```
var $dolly = el.clone();

// make changes to the copy

el.replaceWith($dolly);
```

Don’t Abuse Computed Styles

```
// nein, nein, nein!!!!
for (var i = 0; i < 100; i++) {
    el[i].style.left = el.offsetLeft + "10px";
    el[i].style.top  = el.offsetTop  + "10px";
}
```

```
// Wunderbar
for (
var left = el.offsetLeft, top = el.offsetTop, i = 0;
i < 100;
i++, top+=10, left+=10) {
  el[i].style.cssText += "; left: " + left +
                         "px; top: " + top + "px;";
}
```

Good Reference Points

http://www.phpied.com/rendering-repaint-reflowrelayout-restyle/

http://www-archive.mozilla.org/newlayout/doc/reflow.html

https://developers.google.com/speed/articles/reflow

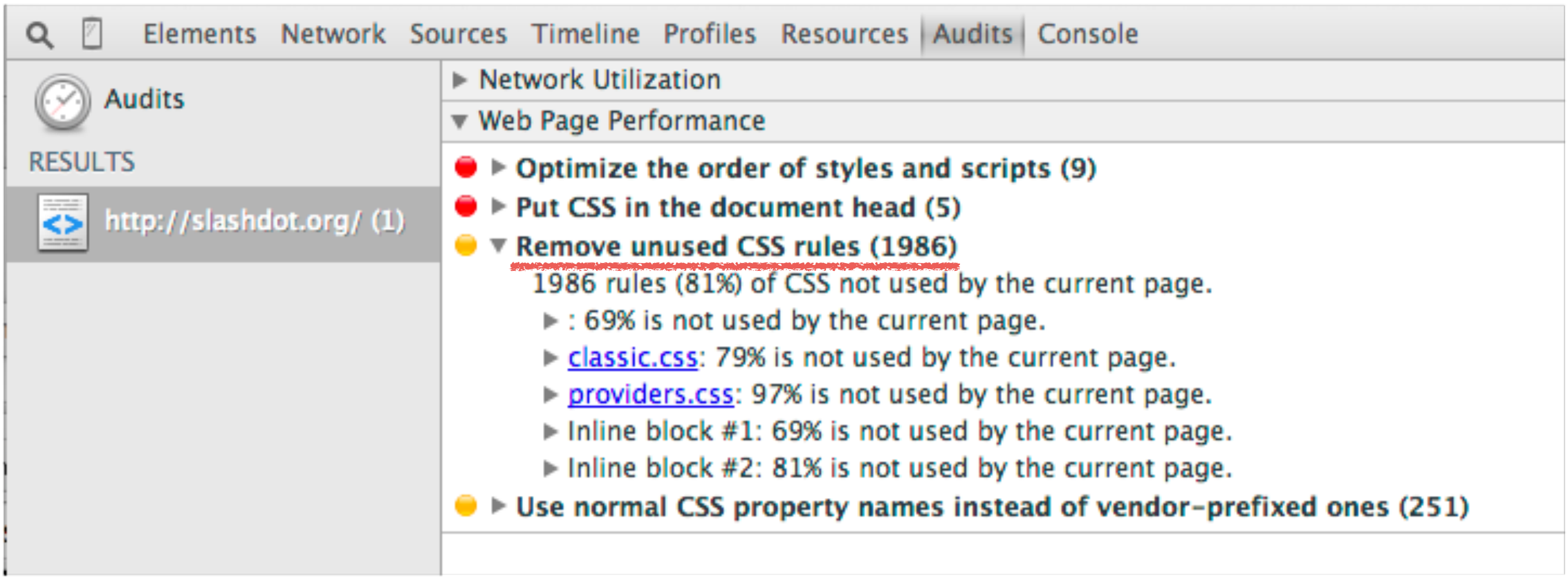
More CSSery

- Reference by element ID
- Be specific, but avoid child selectors
- Avoid @import()
- Avoid multi-class css rule (.foo.bar.baz)

More CSSery

- Pseudo selectors are slow
- Name space attribute selectors (type="…" vs input[type="…"])
- Eliminate un-used rules
- Avoid browser specific extensions (-webkit, -opera, -moz, etc…)

Micro-Case Study: SlashDot.org



CSS Tools

https://github.com/Cerdic/CSSTidy **PHP**

http://devilo.us/ **Web-based**

THANK YOU FOR LISTENING

Slides & Feedback

http://joind.in/14232

http://ilia.ws **@iliaa**