BUILDING A LEAN, MEAN DRUPAL THEME

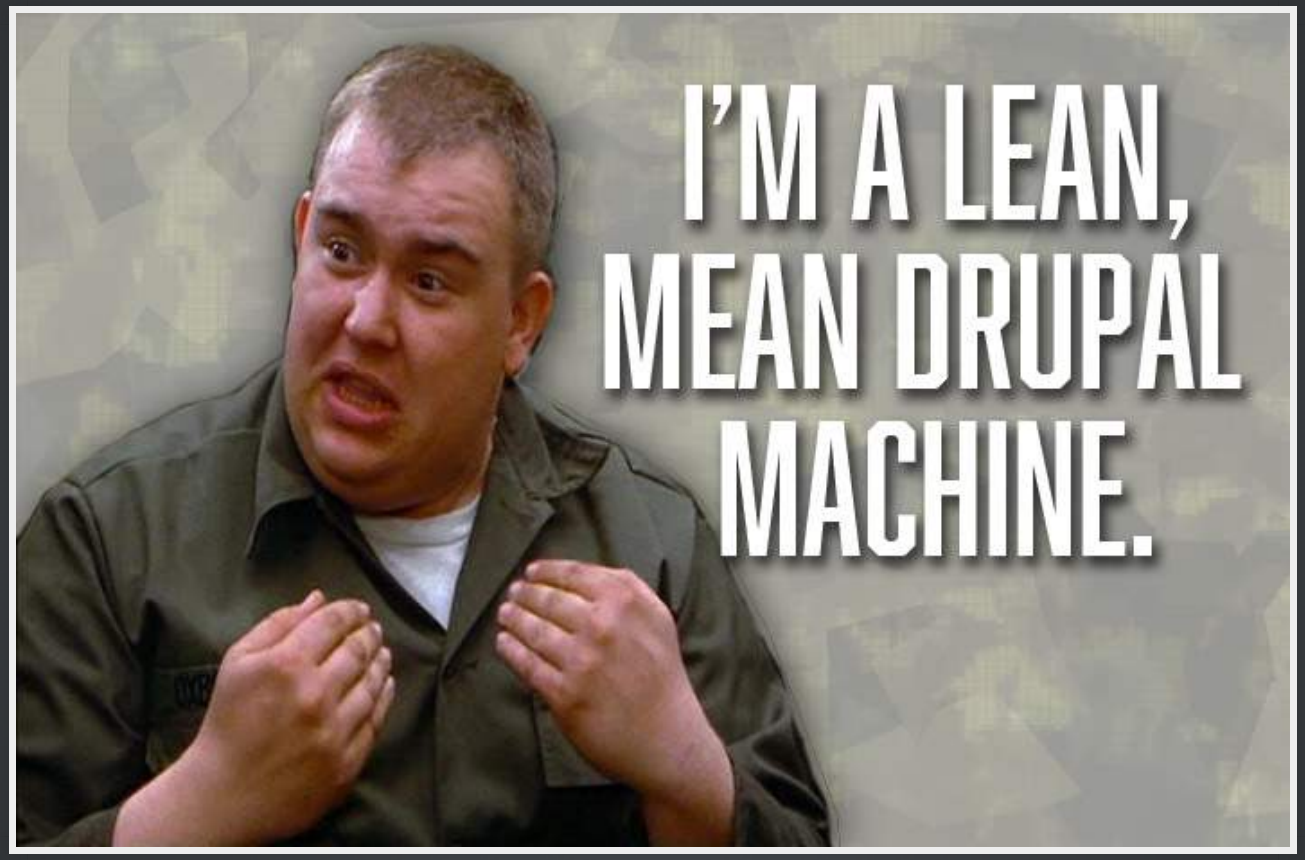
Presented:
July 23, 2015 at DrupalGovCon
August 15, 2015 at Drupal Camp Asheville

JIM SMITH

- Oak Ridge, Tenn.
- Front-end Drupal Developer at DSFederal
- Started using Drupal in 2005
- Drupal user #16880

GET THE HOME VERSION FOR HOURS AND HOURS OF FUN!

http://startinggravity.github.io/lean-drupal


I'M A LEAN,
MEAN DRUPAL
MACHINE.

WHY DO WE CARE ABOUT PERFORMANCE?

- Fast sites make impatient users happier.
- Happy users view more pages.
- Happy users buy more stuff.
- Fast sites are cheaper to host.
- Clean code is easier to maintain.


HOW MUCH DOES THE AVERAGE WEB PAGE WEIGH?

SIZE OF AVERAGE WEB PAGE

(top 1000 websites)

- More than 1600K

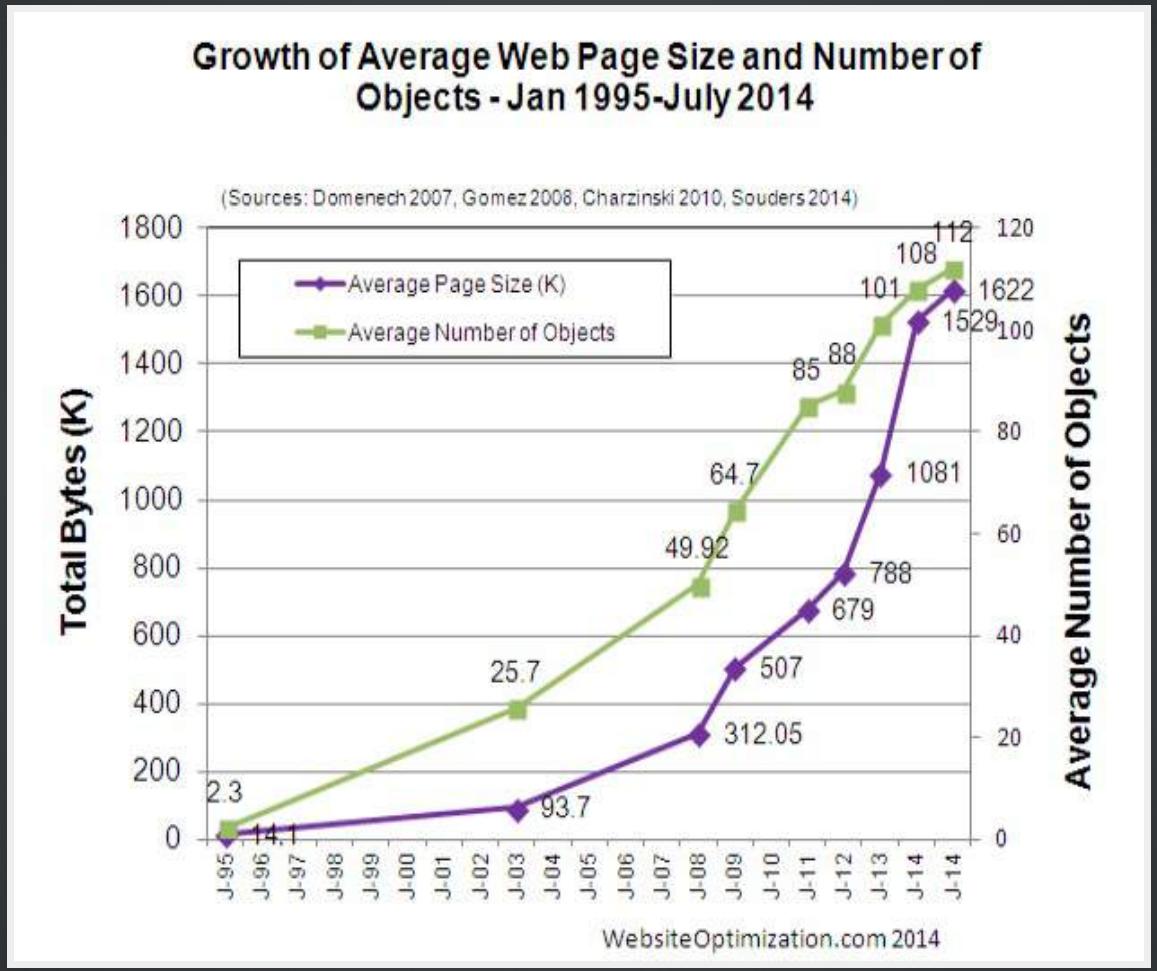
source: WebsiteOptimization.com (July 2014)

NUMBER OF OBJECTS IN AVERAGE WEB PAGE

(top 1000 websites)

- About 112 objects

source: WebsiteOptimization.com (July 2014)



source: WebsiteOptimization.com (July 2014)

Just because we think we have a lot of bandwidth doesn't mean we must use it.

source: Micha Godbolt (@michagodbolt) - July 20, 2015

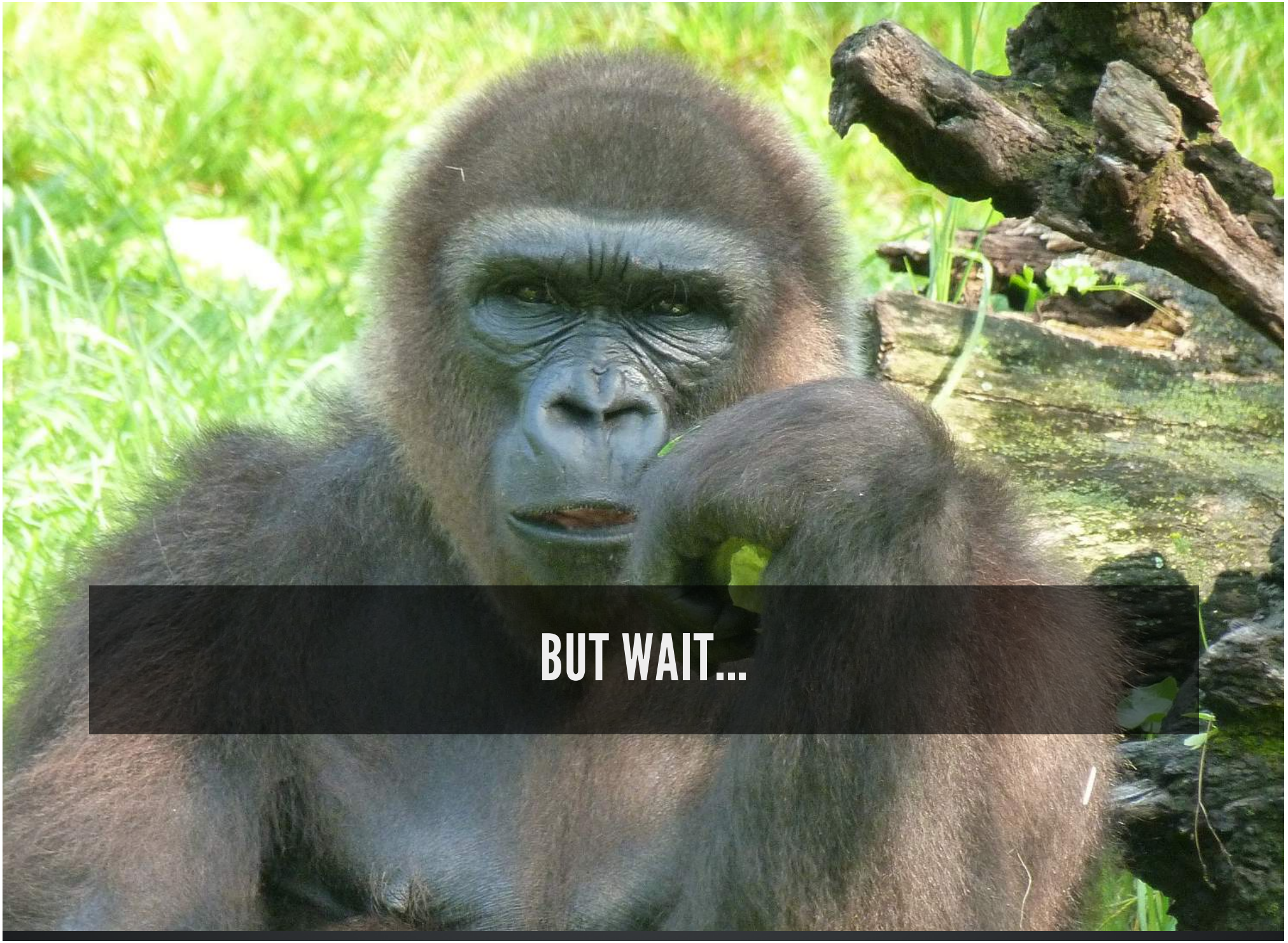
DELAYING PAGE LOAD TIME BY ONE SECOND

- Conversions drop by 7%
- Page views drop by 11%
- Customer satisfaction drops by 16%

source: Aberdeen Group (November 2008)

DON’T BE SLOW.


HOW DO WE GO FASTER?

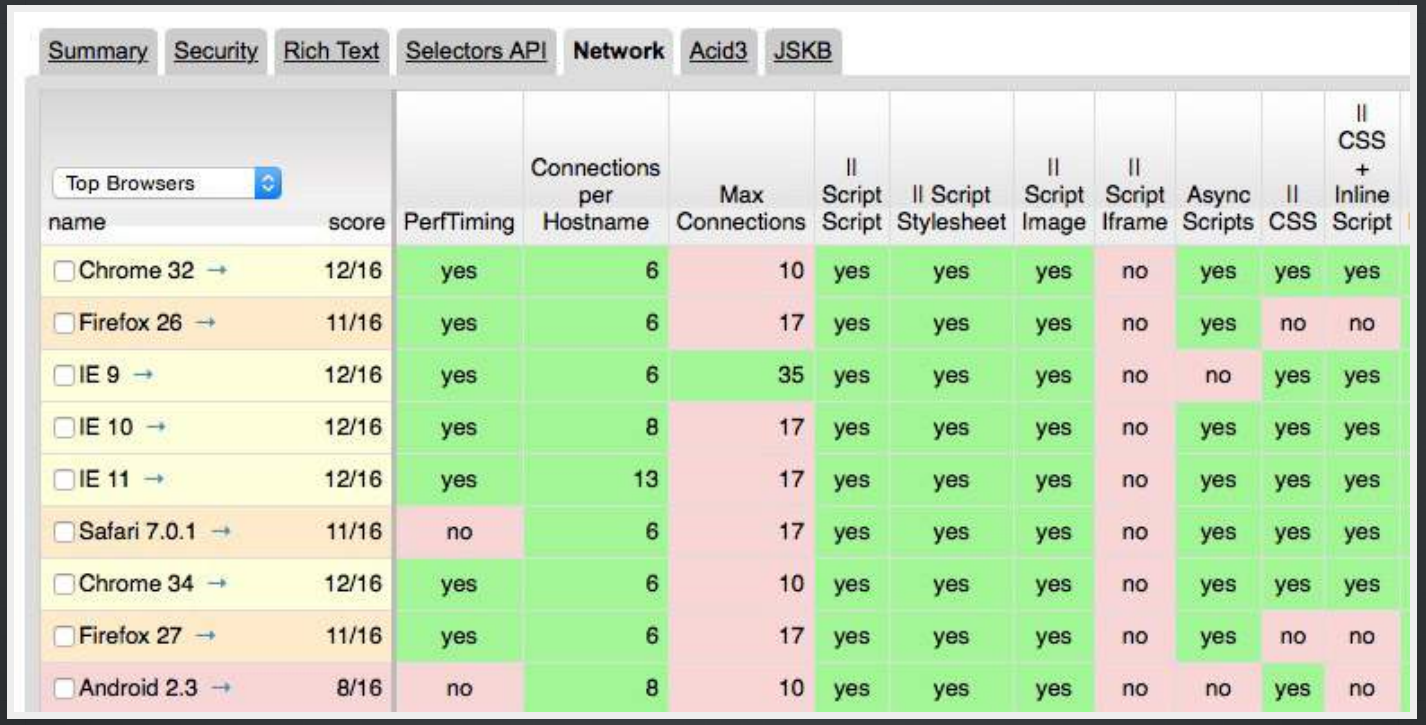

BUT WAIT...

ASSUMPTIONS

- Every site will have different requirements, different libraries, different needs.
- The fastest request you will ever make is the one you don’t make.

LIMITS TO CONNECTIONS

Connectionss per Hostname and Total Connections



Summary | Security | Rich Text | Selectors API | **Network** | Acid3 | JSKB

Top Browsers

name	score	PerfTiming	Connections per Hostname	Max Connections	|| Script Script	|| Script Stylesheet	|| Script Image	|| Script Iframe	Async Scripts	|| CSS	|| CSS + Inline Script
Chrome 32 →	12/16	yes	6	10	yes	yes	yes	no	yes	yes	yes
Firefox 26 →	11/16	yes	6	17	yes	yes	yes	no	yes	no	no
IE 9 →	12/16	yes	6	35	yes	yes	yes	no	no	yes	yes
IE 10 →	12/16	yes	8	17	yes	yes	yes	no	yes	yes	yes
IE 11 →	12/16	yes	13	17	yes	yes	yes	no	yes	yes	yes
Safari 7.0.1 →	11/16	no	6	17	yes	yes	yes	no	yes	yes	yes
Chrome 34 →	12/16	yes	6	10	yes	yes	yes	no	yes	yes	yes
Firefox 27 →	11/16	yes	6	17	yes	yes	yes	no	yes	no	no
Android 2.3 →	8/16	no	8	10	yes	yes	yes	no	no	yes	no

source: Browserscope.com

20% RULE

For a performance change to be noticeable, it must be at least 20% faster than your previous performance.

source: apmblog.dynatrace.com

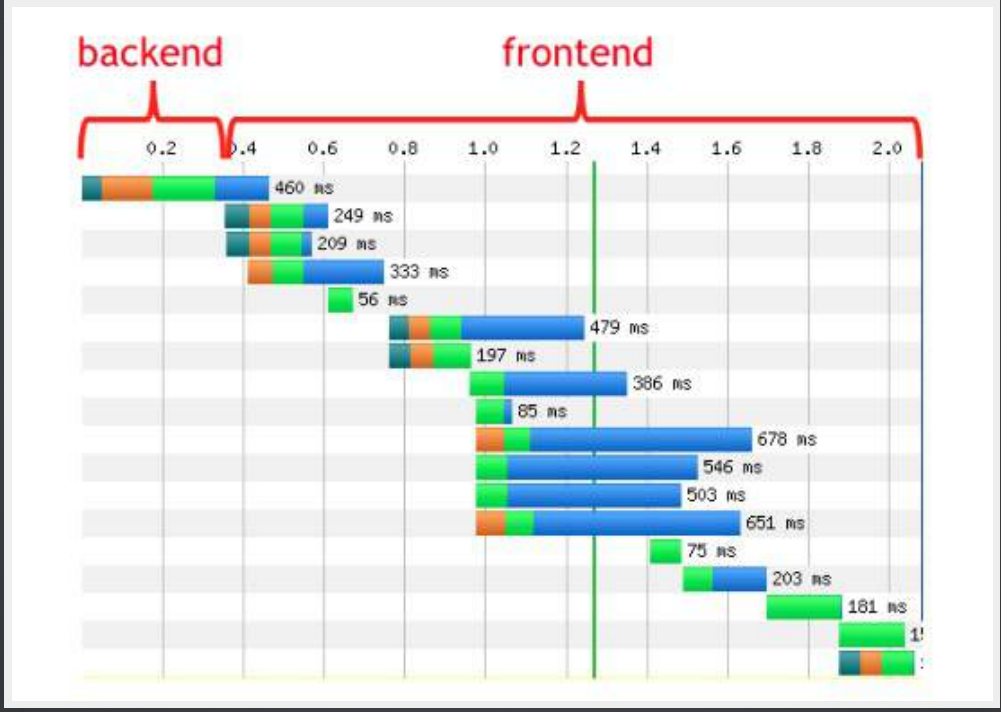
PERFORMANCE GOLDEN RULE

80-90% of the end-user response time
is spent on the frontend.

Start there.

source: Steve Souders Blog

BACKEND VS. FRONTEND



source: Steve Souders Blog

BACKEND VS. FRONTEND

- Backend: The time it takes the server to send the first byte to the browser.
- Frontend: Everything else.

source: Steve Souders Blog

6 WAYS TO SPEED UP YOUR SITE

1. Cache everything you can
2. Clean up Drupal's cruft
3. Clean up your code
4. Concatenate and minify your code
5. Deliver assets efficiently
6. Set a performance budget and test against it


SOME OF THIS INFORMATION WILL BE
RENDERED OBSOLETE BY HTTP/2 AND DRUPAL 8!

CACHE EVERYTHING YOU CAN

"The first rule of optimization and caching is this: never do something time consuming twice if you can hold onto the results and re-use them."

source: Jeff Eaton (Lullabot Blog)

CACHE EVERYTHING YOU CAN

- Turn on page and block caching
- Turn on Views caching, including blocks
- Install Views Content Cache module
- For smaller sites, install a caching module like Boost

CACHE EVERYTHING YOU CAN

- Larger sites should use Memcache or Varnish (must be installed on your server)

CLEAN UP DRUPAL'S CRUFT

"There are a bazillion reasons why Drupal is slow, but the main one is *because you weren't paying attention*."

source: Dan Kegel's Web Hostel

CLEAN UP DRUPAL'S CRUFT

- Don't use core's Update module
- Don't use core's Statistics module
- Use Fast 404 module
- Turn off, and if possible, remove every unnecessary module
- Don't install a module unless there's no way to avoid it

CLEAN UP YOUR CODE

source: Mario Hernandez (Mediacurent Blog)

CLEAN UP YOUR CODE

- Follow Drupal's best practices for themes
- Use a base theme that is inherently clean of unnecessary <divs> and classes
- Use custom theme template files to remove unnecessary stuff
- Use Fences module
- Better yet, create a custom theme without a base theme

CLEAN UP YOUR CODE

- Use BEM, SMACSS and/or OOCSS to organize your CSS
- Understand specificity in your CSS and avoid its traps
- Understand selector efficiency

CONCATENATE AND MINIFY YOUR CODE

"There is a reason jQuery calls the minified version the production version and the original source the development version."

source: Matt Farina (The Engineered Web)

CONCATENATE AND MINIFY YOUR CODE

- Use core's CSS and JS aggregration
- Use Advanced CSS/JS Aggregation (AdvAgg) module
- Make Modernizr more efficient with Modernizr module
- Use Uglifyjs module

CONCATENATE AND MINIFY YOUR CODE

- Use Gulp and gulp-uglify
- Use SVG images when you can, instead of JPG or PNG
- Use software or a service like Smush-it to compress images
- Use Gulp-imagemin to compress images used in your theme during development

DELIVER ASSETS EFFICIENTLY

"There is real empirical evidence that substantiates the fact that speed is more than a feature. It’s a requirement."

source: Fred Wilson "10 Golden Principles of Successful Web Apps"

DELIVER ASSETS EFFICIENTLY

- Move scripts to the footer with Magic module
- Make images responsive and efficient with Picture module
- Distribute assets with CDN module
- Defer image loading with Image Lazyloader module
- Use the Image API Optimize module
- Try a sandbox module to make speculative requests that prefetch and prerender content your visitor is likely to see next.

DELIVER ASSETS EFFICIENTLY

- Instead of a module, just add prefetch links to your theme
- Try domain sharding
- Add inline CSS and JS that's used "above the fold"
- Use the Critical node module with Gulp to automate inline CSS and JS
- Use hosted versions of libraries, such as Google Hosted Libraries

BUDGET AND TEST

"When I first heard the concept of a performance budget, I groaned quietly, rolled my eyes, and thought, 'Oh, great. One more technical thing to stand in my way.'"

source: Katie Kovalcin (Happy Cog Blog)

BUDGET AND TEST

- A performance budget is a goal you set for load times on your site.
- Test representative pages of your site. Do this frequently and consistently.
- Also run tests against competitor or similar sites.
- A budget helps you make decisions on what and how things are displayed.

HOW THE PERFORMANCE BUDGET AFFECTS DEVELOPMENT DECISIONS

- You may be forced to optimize existing content.
- Or remove something no longer important.
- Or leave out a feature that breaks the budget.

4 TYPES OF BUDGETS

We can view metrics in four different ways and measure them accordingly.

- Milestone Timings
- SpeedIndex
- Quantity-based metrics
- Rule-based metrics

source: Tim Kadlec Blog

MILESTONE TIMINGS BUDGET

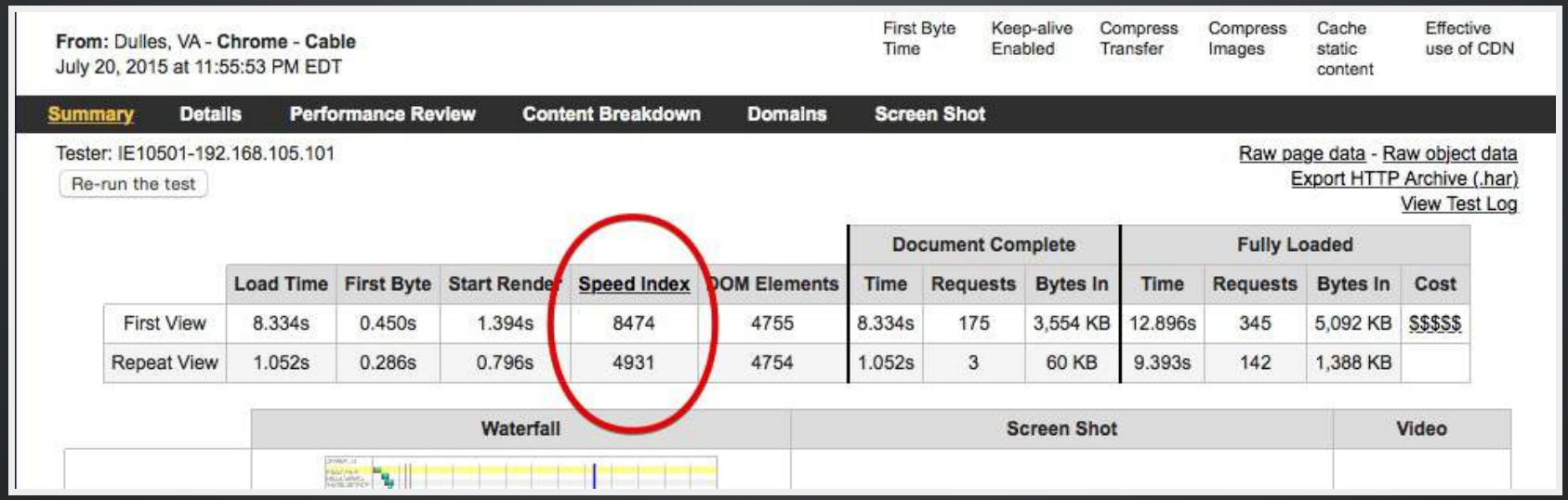
Typically, the time to render a page

- Measures the whole page, not just what the visitor sees
- Useful if you set your own milestone, such as time to expose a form

SPEEDINDEX

The average time it takes for visible parts of a page to display, expressed in milliseconds and dependent on size of the viewport.

- An effective test because it measures what a user sees
- Best measured with WebPageTest.org



From: Dulles, VA - **Chrome** - **Cable**
July 20, 2015 at 11:55:53 PM EDT

First Byte Time | Keep-alive Enabled | Compress Transfer | Compress Images | Cache static content | Effective use of CDN

Summary | **Details** | **Performance Review** | **Content Breakdown** | **Domains** | **Screen Shot**

Tester: IE10501-192.168.105.101
Re-run the test

Raw page data - Raw object data
Export HTTP Archive (.har)
View Test Log

						Document Complete			Fully Loaded			
	Load Time	First Byte	Start Render	Speed Index	DOM Elements	Time	Requests	Bytes In	Time	Requests	Bytes In	Cost
First View	8.334s	0.450s	1.394s	8474	4755	8.334s	175	3,554 KB	12.896s	345	5,092 KB	$$$$$
Repeat View	1.052s	0.286s	0.796s	4931	4754	1.052s	3	60 KB	9.393s	142	1,388 KB	

Waterfall | Screen Shot | Video

QUANTITY-BASED METRICS

A simple counting of all requests issued or the total weight of a page.

- Easy to measure and easy to track during development
- Tells you nothing about user experience

RULE-BASED METRICS

More a checklist of optimizations you should be doing.

- Can be used as a metric in a budget
- Use Chrome PageSpeed Insights or YSlow Scores


Thank You!

Find this presentation at:
http://startinggravity.github.io/lean-drupal

Find me at:

- Email: jim.smith@dsfederal.com
- Twitter: @_JimSmith_
- IRC: startinggravity