



Web Application Performance From User Perspective



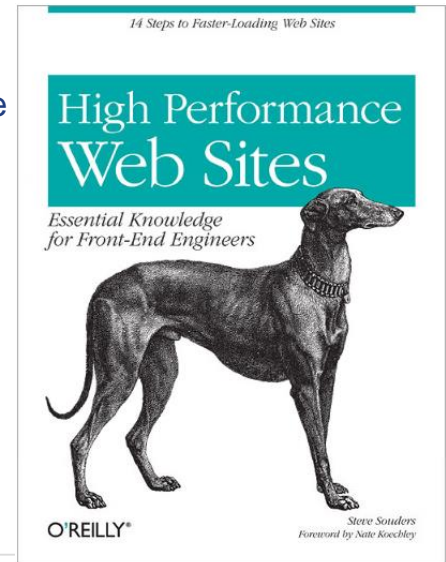
AGENDA

1. Tips for improving web page performance
2. Metrics
3. Measurements techniques

- My foundation of technical expertise stems from hands on developing, testing along with team leading.
- As consultant I was a part of various test teams and was involved in testing on all software life cycle stages. I build extensive experience while testing critical functions of real time system which is GSM, financial platforms and also web and desktop applications.
- Currently I supervise testing activities in one of the project for GFT Technologies

Tips For High Performance Web Sites

- Gzip Components
- Put StyleSheets at the Top
- Put Scripts at the Bottom
- Minimize HTTP Requests
- Minify JavaScript and CSS
- Make JavaScript and CSS External
- Use a Content Delivery Network
- Avoid empty src or href
- Add an Expires or a Cache-Control Header
- Avoid CSS Expressions
- Reduce DNS Lookups
- Remove Duplicate Scripts
- Make AJAX Cacheable
- Use GET for AJAX Requests
- Reduce the Number of DOM Elements
- Use Cookie-Free Domains for Components
- Do Not Scale Images in HTML
- Make favicon.ico Small and Cacheable
- Configure ETags
- Avoid Redirects
- Reduce Cookie Size
- Avoid Filters
- No 404s



Checking conformance to High Performance Web Sites Rules

- YSlow
- Google PageSpeed

The screenshot shows the YSlow Chrome extension interface. The browser address bar displays the URL: chrome-extension://ninejjcohidippngpapii...mkglImakh/yslow.html#48. The extension's navigation bar includes 'Home', 'Grade', 'Components', and 'Statistics'. The 'Grade' tab is active, showing an overall performance score of 72. The ruleset applied is 'YSlow(v2)' and the URL is 'http://www.ishares.com/uk/?...'. There are 23 rules in total, categorized by filter: CONTENT (6), COOKIE (2), CSS (6), IMAGES (2), JAVASCRIPT (4), and SERVER (6). A detailed view of the 'Grade F on Make fewer HTTP requests' rule is shown on the right, explaining that the page has 20 external JavaScript scripts, 3 external stylesheets, and 13 external background images, and providing suggestions for optimization.

chrome-extension://ninejjcohidippngpapii...mkglImakh/yslow.html#48

Home Grade Components Statistics Rulesets YSlow(v2) Edit Help

Grade Overall performance score 72 Ruleset applied: YSlow(v2) URL: http://www.ishares.com/uk/?...

ALL (23) FILTER BY: CONTENT (6) COOKIE (2) CSS (6) IMAGES (2) JAVASCRIPT (4) SERVER (6) Tweet Share

F Make fewer HTTP requests

- F Use a Content Delivery Network (CDN)
- A Avoid empty src or href
- F Add Expires headers
- D Compress components with gzip
- A Put CSS at top
- A Put JavaScript at bottom
- A Avoid CSS expressions
- n/a Make JavaScript and CSS external
- D Reduce DNS lookups
- A Minify JavaScript and CSS
- A Avoid URL redirects
- A Remove duplicate JavaScript and CSS
- C Configure entity tags (ETags)
- A Make AJAX cacheable
- A Use GET for AJAX requests
- F Reduce the number of DOM elements
- A Avoid HTTP 404 (Not Found) error
- A Reduce cookie size
- F Use cookie-free domains
- A Avoid AlphaImageLoader filter
- A Do not scale images in HTML
- A Make favicon small and cacheable

Grade F on Make fewer HTTP requests

This page has 20 external Javascript scripts. Try combining them into one.
This page has 3 external stylesheets. Try combining them into one.
This page has 13 external background images. Try combining them with CSS sprites.

Decreasing the number of components on a page reduces the number of HTTP requests required to render the page, resulting in faster page loads. Some ways to reduce the number of components include: combine files, combine multiple scripts into one script, combine multiple CSS files into one style sheet, and use CSS Sprites and image maps.

[»Read More](#)

Copyright © 2015 Yahoo! Inc. All rights reserved.

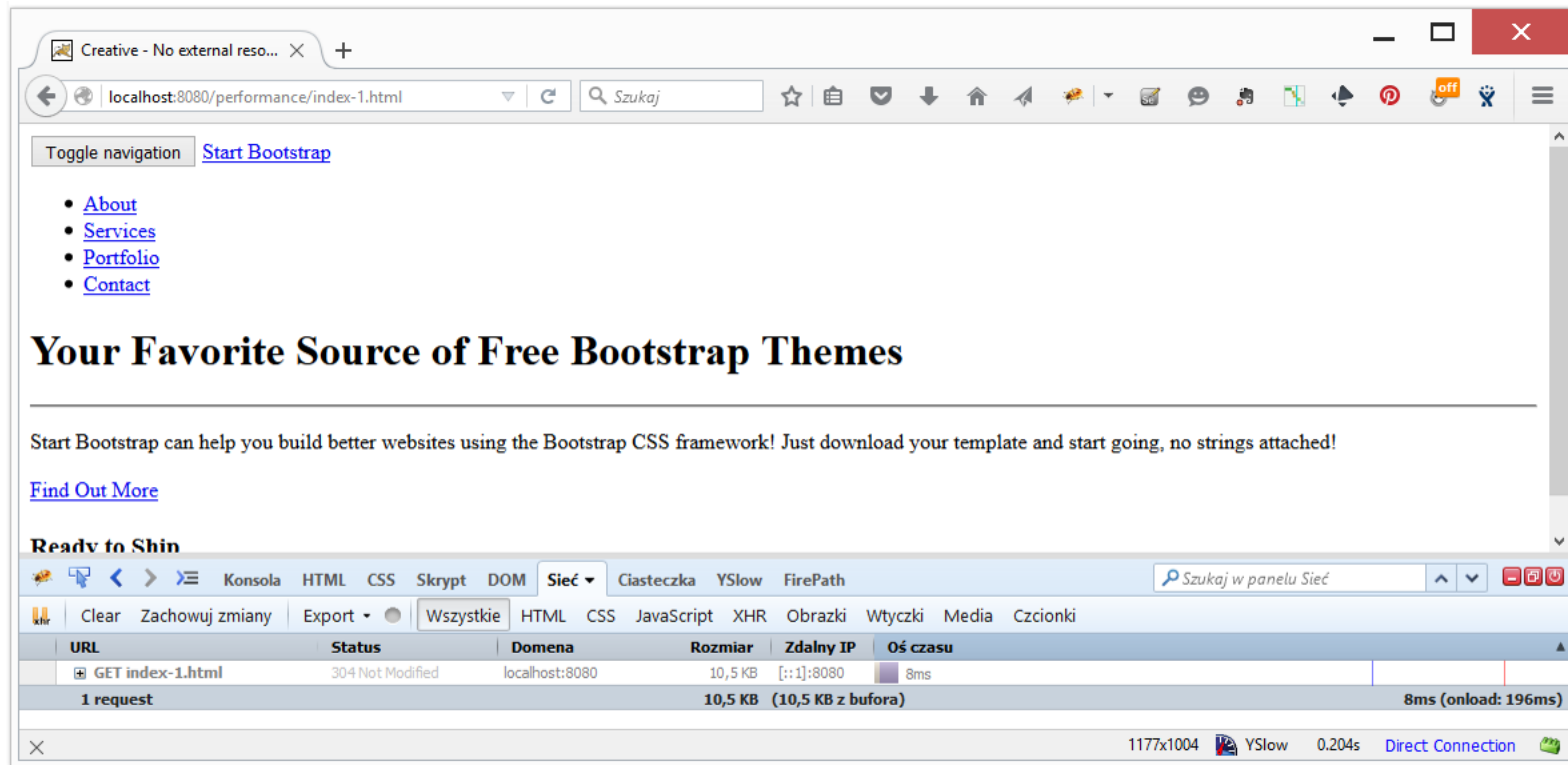
Performance testing with load generators

- Overview
 - Performed with tools like Jmeter, Gatling, AB, LoadRunner etc.
 - Generates concurrent requests based on scripted instruction
 - Provides metrics based on response time for request

- Advantages
 - Requires relatively small amount of hardware for simulating huge amount of concurrent users
 - Loads server resources (CPU, memory) and network
 - Easy to run and collect results

- Disadvantages
 - Does not work exactly as browsers

Browser vs load generator results



The screenshot shows a web browser window with the address bar displaying `localhost:8080/performance/index-1.html`. The page content includes a navigation menu with links for [About](#), [Services](#), [Portfolio](#), and [Contact](#). The main heading is

Your Favorite Source of Free Bootstrap Themes

, followed by a paragraph: "Start Bootstrap can help you build better websites using the Bootstrap CSS framework! Just download your template and start going, no strings attached!" and a link [Find Out More](#).

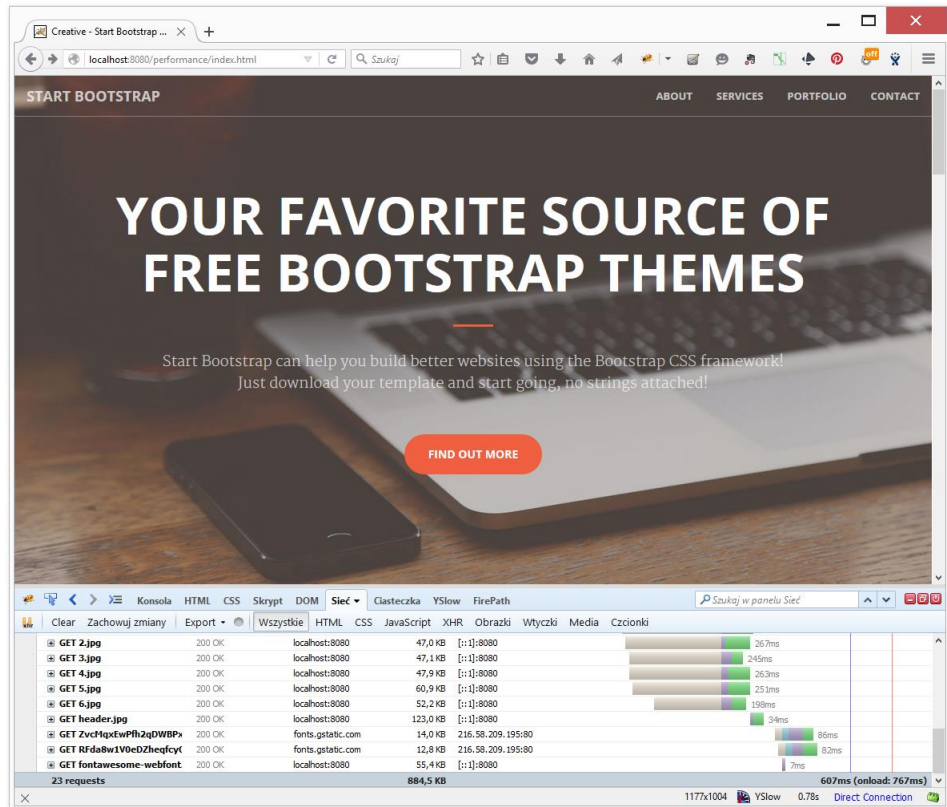
The browser's developer tools are open to the Network tab, showing a single request:

URL	Status	Domena	Rozmiar	Zdalny IP	Oś czasu
GET index-1.html	304 Not Modified	localhost:8080	10,5 KB	:::1]:8080	8ms
1 request			10,5 KB	(10,5 KB z bufora)	8ms (onload: 196ms)

At the bottom of the developer tools, the connection details are shown: `1177x1004`, `YSlow`, `0.204s`, and `Direct Connection`.

USING REAL BROWSER IN PERFORMANCE TESTS

- Key advantages
 - Provides metrics from real browser
 - You can run them when you want
- Key disadvantage
 - User perspective as if she use test equipment browser, hardware, connections speed, location)
- Key challenges
 - Where to put test equipment
 - What metrics should be collected

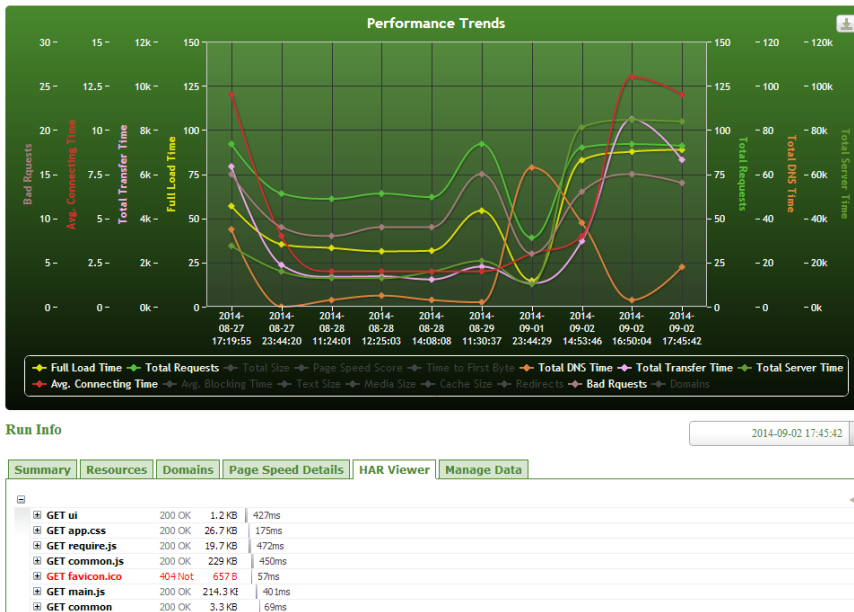


■ Open Source

- Selenium with BrowserMob and HarStorage – simple on site solution
- SiteSpeed.io – tool that can crawl part or entire web site and collect yslow and performance measurements
- WebPageTest – collect measurements from various locations, browsers and connection seeds. It captures filmstrips, allows for scripting, heavy customization and calling via API

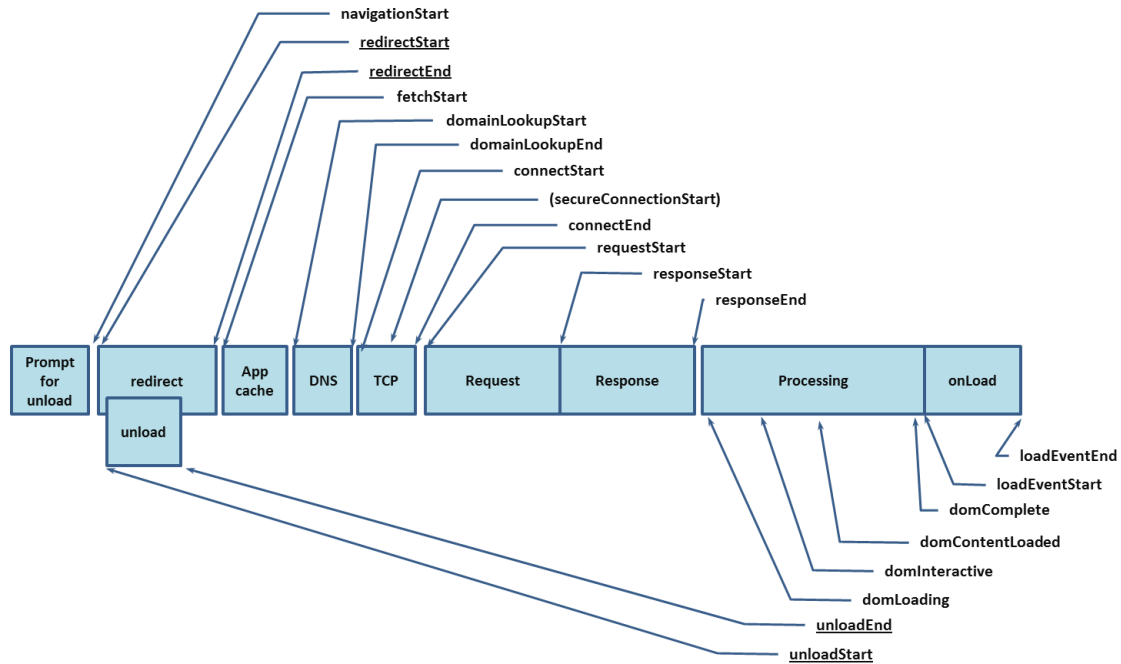
■ Commercial services

- NewRelic – affordable tool that can be used to synthetic measurements from various locations (AWS) and RUM
- CatchPoint – gives you give insight into application performance from lots of locations around the world



Browser events

- Common metrics
 - domContentLoaded
 - load (loadEventEnd)

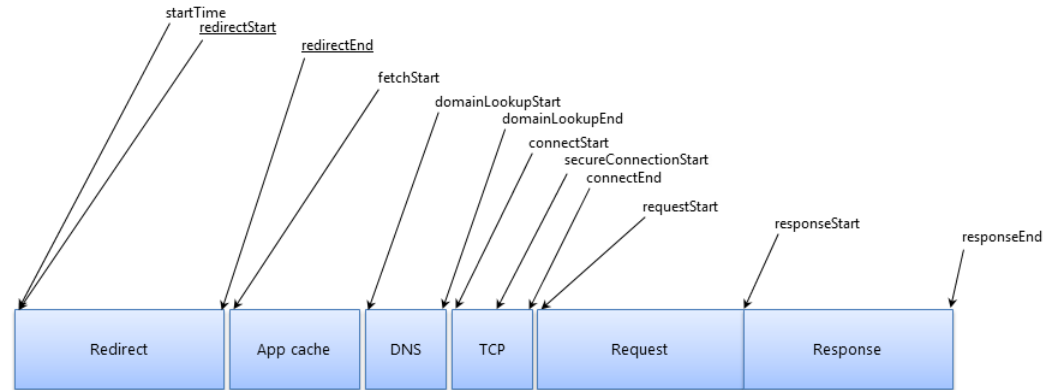


Navigation Timing API

```
var performance = window.performance ||  
    window.webkitPerformance || window.mozPerformance ||  
    window.msPerformance || {};  
var t = performance.timing || {};  
var interactive = t.domInteractive - t.navigationStart,  
    dcl = t.domContentLoadedEventStart - t.navigationStart,  
    complete = t.domComplete - t.navigationStart;  
return interactive + ', ' + dcl + ', ' + complete;
```

Resources

- Common metrics
 - Specific resource loaded
 - Last resource loaded



Custom measurements

- Common metric
 - Specific resource displayed

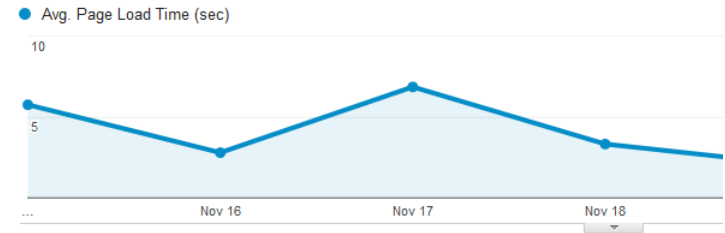
```
<link rel="stylesheet" type="text/css"
      media="all"
      href="contentSearchUI.css">
...

<script>performance.mark('hero2')</script>
```

* https://www.youtube.com/watch?v=f5_iAzS3WMQ

Real User Measurements (RUM)

- Collected from real user devices (mobile and desktop) and helps understand how performance is seen by real users
- Key advantages
 - You know what performance is seen by real users
 - Solves most of the challenges of synthetic monitoring
- Key disadvantage
 - You can not run it when you want (for example before release)
- Key challenges
 - What to measure



Primary Dimension: Page Other

Plot Rows Secondary dimension: Country Sort Type: Default

Page ?	Country ?	Avg. Page Load Time (sec) ↓
		3.42 Avg for View: 2.95 (16.09%)
1. /	Japan	10.10
2. /	Israel	5.75
3. /	Belgium	5.14
4. /	Switzerland	4.65
5. /	Germany	3.63
6. /	United Kingdom	2.69
7. /	Gibraltar	2.67

References

- <https://www.soasta.com/blog/measuring-web-performance-video/>
- <https://dzone.com/articles/a-brief-history-of-web-performance-roi>
- <https://developers.google.com/web/fundamentals/performance>
- <http://yslow.org/>
- <http://www.softwareishard.com/blog/har-viewer/>
- <https://www.soasta.com/blog/ebook-usertiming-performance-monitoring/>
- <https://codeascraft.com/>
- <https://www.soasta.com/blog/ebook-usertiming-performance-monitoring/>
- <https://code.facebook.com/posts/991252547593574/the-technology-behind-preview-photos/>
- <https://blog.twitter.com/2012/improving-performance-on-twittercom>
- <http://www.stevesouders.com/blog/2015/05/12/hero-image-custom-metrics>



Steve Souders
@Souders



Ilya Grigorik
@igrigorik



Tammy Everts
@tameverts

Thank you

GFT Technologies SE
Jacek Okrojek
Email: jacek.okrojek@gft.com

