Τεχνολογίες Εφαρμογών Ιστού

Website Performance Analysis and Optimization

Βασισμένο στη διπλωματική του φοιτητή Αναστάσιου Δρόσου

Why performance

- Slower website equals lower conversion rates and company profits
  - Google: 400 milliseconds delay – 0,59% drop in searches
  - Yahoo: 400 milliseconds delay – 5% until 9% drop in full page traffic
  - Amazon: 1.6$ billion per year if website becomes 1 second slower

User’s behavior

- 47% expect a website to load on 2 or less seconds
- 40% will abandon after waiting for 3 seconds
- 52% claim that speed is important for the loyalty of the website
- 64% that have a bad experience, will visit another website next time
- 88% are less likely to return after a bad experience
- More than a third told others about bad experience

Speed matters

- Google decided to take speed into account in the search rankings
- Google metrics are:
  - How a page responds to Google Bot
  - Load time of website

Web Site Principles

- Availability
- Performance
- Reliability
- Scalability
- Manageability
- Cost

Monitoring tools

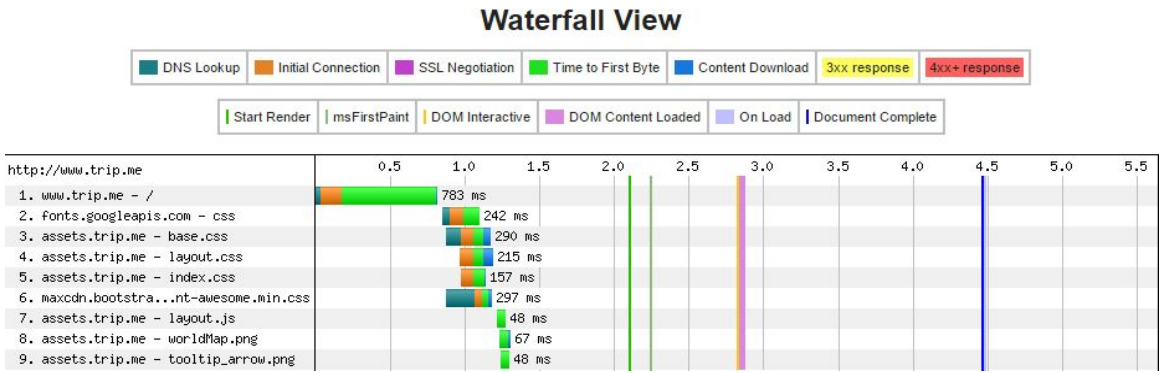
- WebPageTest
- GTmetrix
- PageSpeed Insights
- Blackfire.io
- Google Analytics

WebPageTest

- Open Source (https://github.com/WPO-Foundation/webpagetest)
- Google maintained it, at least until recently

							Document Complete			Fully Loaded		
Load Time	First Byte	Start Render	Visually Complete	Speed Index	DOM Elements	Result (error code)	Time	Requests	Bytes In	Time	Requests	Bytes In
4.461s	0.800s	2.098s	5.192s	2409	926	0	4.461s	76	1,703 KB	5.627s	87	1,816 KB

RUM First Paint	domInteractive	domContentLoaded	loadEvent
2.236s	2.821s	2.821s - 2.870s (0.049s)	4.456s - 4.485s (0.029s)



WebPageTest

Full Optimization Checklist

http://trip.me	Keep-Alive 100%	GZip 97%	Compress Img 100%	Progressive 0%	Cache Static 79%	CDN Detected 92%
1: trip.me - /					✗	
2: www.trip.me - /	✓	✓			✗	
3: fonts.googleapis.com - css		✓			✗	✓
4: maxcdn.bootstrap...ont-awesome.min.css	✓	✓			✓	✓
5: assets.trip.me - base.css	✓	✓			✓	✓
6: assets.trip.me - layout.css	✓	✓			✓	✓
7: assets.trip.me - index.css	✓	✓			✓	✓
8: assets.trip.me - layout.js	✓	✓			✓	✓
9: assets.trip.me - worldMap.png	✓		✓		✓	✓
10: assets.trip.me - tooltip_arrow.png	✓		✓		✓	✓
11: assets.trip.me - sprites.png	✓		✓		✓	✓
12: fonts.gstatic.c...IZOnv9q090hN8.woff2	✓				✓	
13: assets.trip.me...price-guaranteed.png	✓		✓		✓	✓
14: assets.trip.me - secure-payment.png	✓		✓		✓	✓
15: fonts.gstatic.c...qr5-oayXSOefg.woff2	✓				✓	
16: fonts.gstatic.c...TICgirnJhmVJw.woff2	✓				✓	
17: assets.trip.me - quality-agencies.png	✓		✓		✓	✓
18: assets.trip.me - trusted-contact.png	✓		✓		✓	✓
19: assets.trip.me - customer-reviews.png	✓		✓		✓	✓
20: assets.trip.me - footer-twitter.png	✓		✓		✓	✓
21: photos.trip.me...54ba33d533554d5.jpeg	✓		✓	✗	✓	✓
22: photos.trip.me...2e925520d76aa38.jpeg	✓		✓	⚠	✓	✓
23: photos.trip.me...ef016f5ef6f3ec3.jpeg	✓		✓	⚠	✓	✓
24: photos.trip.me...bc906a08cbddb73.jpeg	✓		✓	⚠	✓	✓
25: photos.trip.me...88fc6e590d0b2be.jpeg	✓		✓	⚠	✓	✓
26: photos.trip.me...6ba677da4e74edc.jpeg	✓		✓	⚠	✓	✓
27: assets.trip.me...ooter-googleplus.png	✓		✓		✓	✓
28: photos.trip.me...332c9336d489472.jpeg	✓		✓	⚠	✓	✓
29: assets.trip.me - footer-emirates.png	✓		✓		✓	✓
30: assets.trip.me - footer-forbes.png	✓		✓		✓	✓
31: photos.trip.me...2237fdd76be1378.jpeg	✓		✓	⚠	✓	✓

GTmetrix

- PageSpeed Score: Same with WebPageTest
- Yslow: Based on Yahoo rules

Latest Performance Report for:

http://trip.me/

Report generated: Sat, Sep 24, 2016, 11:42 AM -0700

Test Server Region: Vancouver, Canada

Using: Firefox (Desktop) 47.0, PageSpeed 1.15-gt1, YSlow 3.1.8

Looks like you might not be using a CDN
Why should I use a CDN? »

Performance Scores

PageSpeed Score	YSlow Score
A (90%) ^	D (63%) v

Page Details

Page Load Time	Total Page Size	Requests
4.2s ^	1.72MB ^	80 v

Blackfire.io

- Product of Symfony and SensioLabs
- Paid usage
- Basic metrics:
  - I/O
  - CPU
  - Memory
  - SQL

Blackfire.io waterfall

#1. WP clean | 203 ms | 6.21 ms | 0 ms / 0 B / 0 rq | 3.24 ms / 22 rq | Replay

Functions | Metrics | Assertions

search

Function / Metric	% Excl.	% Incl.	Calls
MO::make_entry			1606

1 caller (1606 calls)

MO
make_entry

24.6 ms
0 µs
24.6 ms
0.983 MB
0 B

3 callees

do_action	46
apply_filters	2323
get_option	308
Translation_Entry::key	2196
POMO_Reader::substr	3214
...Widget::display_callback	6
do_action@1	17

main()

load_default_textdomain
32.28%

run_init():
web/wp-settings.php
62.86%

load_default_textdomain
32.28%

load_textdomain
32.27%

MO::
import_from_file
32.16%

MO::
import_from_reader
32.07%

MO::
make_entry
12.11%

Compiled files
2.80%

create_initial_post_types
5.48%

get_sid
9.03%

locate_template()
9.02%

1 x
7 x
1 x
1 x
1 x
54 x
1 x
12 x
1 x
1606 x

Feedback

Frontend Optimization

- Content
- Server
- CSS
- JavaScript
- Images

Content

- Make fewer requests
  - Higher number or requests, slower response time
  - Merge files and images
- Reduce DNS Lookups
- Split components across domains
  - Firefox can do 6 parallel requests by default

Server

- CDN (Content Delivery Network)
- Caching
- Compression

Content Delivery Network

- Redirection of requests to a server that is geographically closer

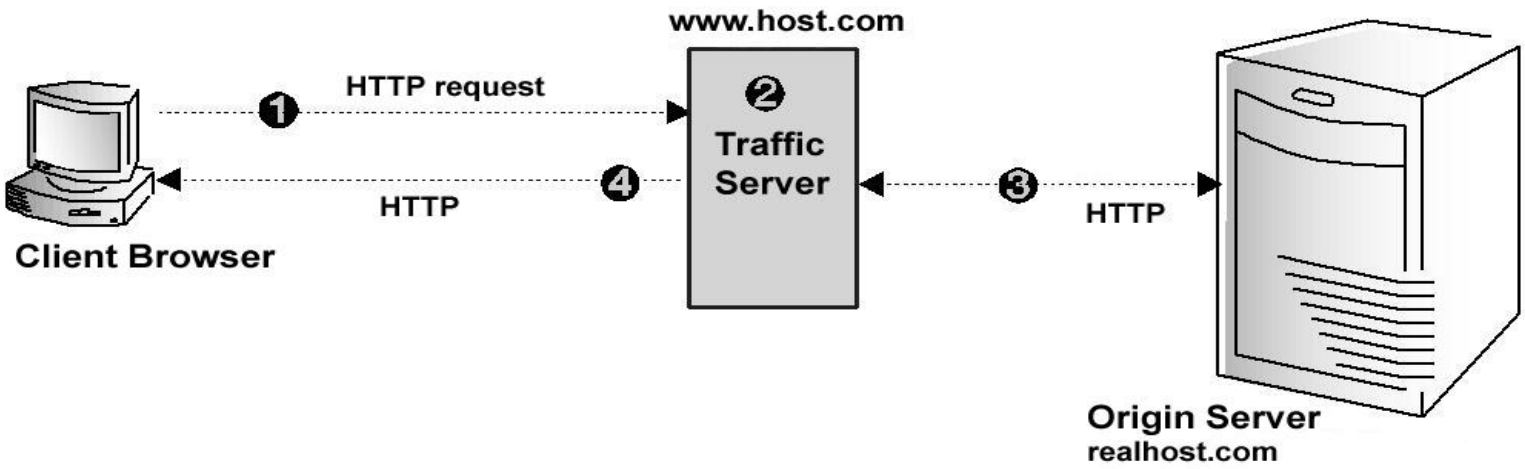
Caching – browser cache

- All browsers have caching settings:
  - The time the information is cached
  - Total amount of information that is cached

- Response served by browser cache:
  - Status: '200 OK (BFCache)'
  - Header: 'The request was resolved directly from the cache, so we have no response from the server. See below for the cached response.'

Caching – reverse proxy

- Sits in front of the application
- It caches files
- It caches whole requests
- It achieves huge redundancy in network traffic



Expiration

- How long files are “fresh”
  - Expires
  - Cache-control: max-age
- Response
  - Expires: Thu, 31 Dec 2015 16:00:00 GMT.
  - Cache-Control: max-age=31536000.
- Images, CSS and JavaScript files should have a far future expiration date

Validation

- It forces revalidation when content of files changes
- Last-modified date
  - Request: If-Modified-Since Mon, 19 Jan 2015 08:03:22 GMT.
  - Response: Date Fri, 06 Mar 2015 14:05:16 GMT.
- Etag
  - Request: If-Modified-Since Mon, 19 Jan 2015 08:03:22 GMT.
  - Response: Date Fri, 06 Mar 2015 14:05:16 GMT.
  - Status: 304 Not Modified

Cache control HTTP headers

- Max-age=[seconds]
- S-max-age=[seconds]
- Public
- Private
- No-cache
- No-store
- Must-revalidate
- Proxy-revalidate

Compression

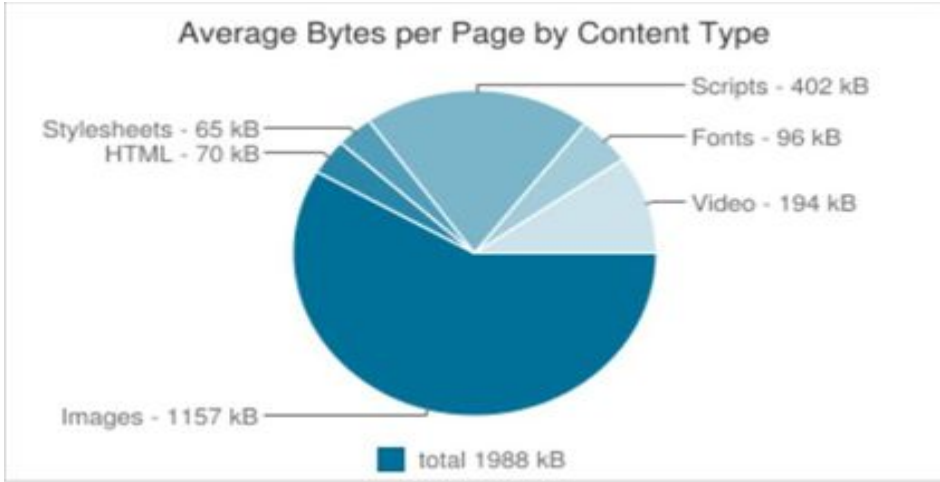
- Gzip compression
- Achieves size reduction up to 70%
- Everything that contains text should be gzipped
- Accept encoding: gzip, deflate
- Content-encoding: gzip
- Mod_deflate on apache

CSS and JavaScript

- Put stylesheets at the top – allow progressive rendering
- Put scripts at the bottom – block parallel download
- Make JavaScript and CSS external (allow caching)
- Remove duplicate scripts
- Minify CSS and JavaScript files (allow faster compiling of files)
  - YUI compressor
  - JSMin compressor

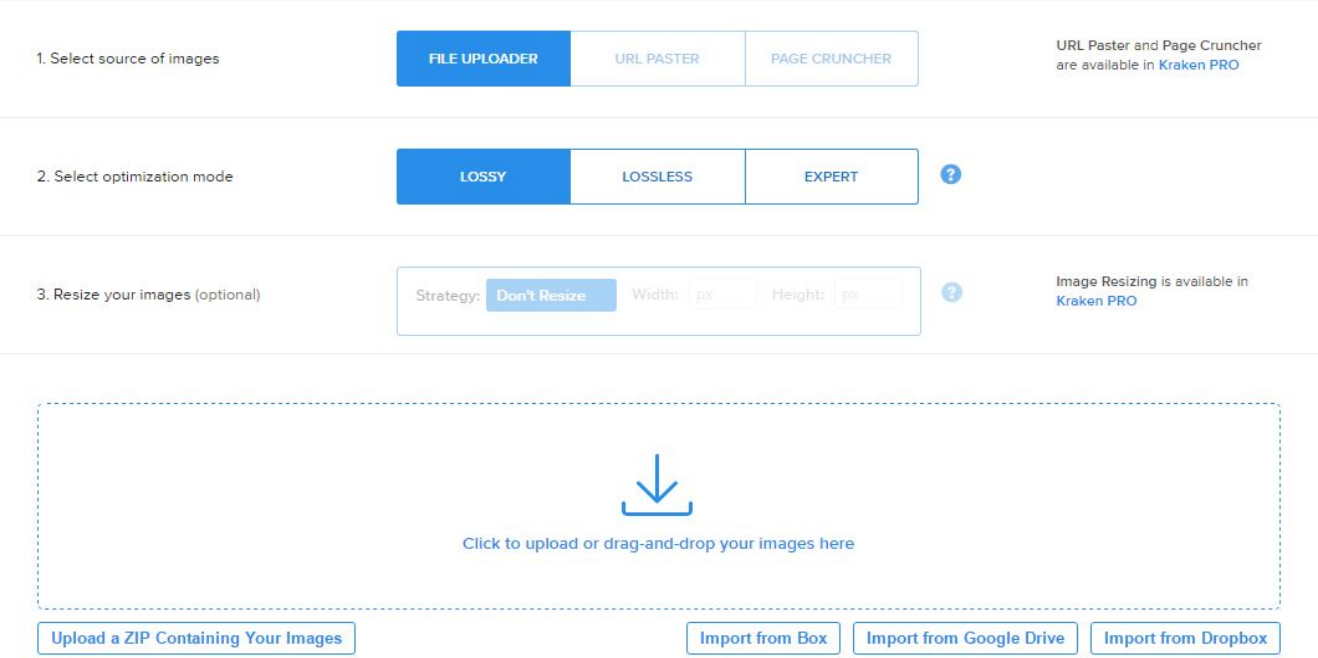
Images

- Image compression
- Lossy vs Lossless
- Do not scale images in HTML
- Make favicon.ico small and cacheable
- Optimize CSS sprites
- Progressive vs Baseline JPEGs



Kraken.io

Online tool



Back-end optimization - Metrics

- Web Server
  - Clean Code
  - Database server

- Metrics:
  - Time to first byte
  - Server response time

Web Server

- It serves http requests
- Dynamic server vs static server
- Most common used web servers
  - Apache
  - Nginx
  - IIS
  - ...nodeJS http-server

Apache rules

- Keep apache up-to-date
- Disable unused modules
  - Perl
  - Python
  - Rack / Ruby / Passenger
  - SSL
  - PHP
- Allocate RAM for Apache
- Allow apache multi processing

Load Balancer

- Sits in front of the servers
- Distributes client requests or network load efficiently across multiple servers
- Ensures high availability and reliability by sending requests only to servers that are online
- Provides the flexibility to add or subtract servers as demand dictates

- Balancing Algorithms
  - Round Robin
  - Least Connections
  - IP Hash

Amazon web service auto scale

- Maintain application availability
- Scale up or down capacity based on needs
- Increase the number of amazon instances based on needs
- Well suited to stable demands
- Well suited to hourly, daily and weekly variability in usage

Clean Code

- Is focused (SRP)
  - Single-responsibility principle
- It should not be redundant (DRY rule)
  - do not repeat yourself
- Reading code should be pleasant (KISS, YAGNI)
  - Keep it simple, stupid
  - you aren't gonna need it
- Can be easily extended by other developers
- It has minimal dependencies (do not break SRP)
- It is small
- It should be supported by tests
- It should be expressive
- Do not contain comments (use TODOs)

PHP mess detector

- It takes a given PHP source code base and look for several potential problems within that source.

- These problems can be things like:
  - Possible bugs
  - Suboptimal code
  - Overcomplicated expressions
  - Unused parameters, methods, properties

PHPMD example

```
mapi@arwen ~ $ phpmd PHP/Depend/DbusUI/ xml rulesets/codesize.xml
<?xml version="1.0" encoding="UTF-8" ?>
<pmd version="0.0.1" timestamp="2009-12-19T22:17:18+01:00">
  <file name="/projects/pdepend/PHP/Depend/DbusUI/ResultPrinter.php">
    <violation beginline="67"
               endline="224"
               rule="TooManyMethods"
               ruleset="Code Size Rules"
               package="PHP_Depend\DbusUI"
               class="PHP_Depend_DbusUI_ResultPrinter"
               priority="3">
      This class has too many methods, consider refactoring it.
    </violation>
  </file>
</pmd>
```

PHP Coding Standard Fixer

- Developed by SensioLabs
- Checks whether code meets the PSR (PHP Standards Recommendations) rules

```
php php-cs-fixer.phar fix /path/to/project --level=psr0
php php-cs-fixer.phar fix /path/to/project --level=psr1
php php-cs-fixer.phar fix /path/to/project --level=psr2
php php-cs-fixer.phar fix /path/to/project --level=symfony
```

PHP Coding Standard Fixer example rules

- PSR0: Classes must be in a path that matches their namespace, be at least one namespace deep, and the class name should match the file name.
- Elseif: The keyword elseif should be used instead of else if so that all control keywords looks like single words.
- Eof-ending: A file must always end with a single empty line feed.
- Function-call-space: When making a method or function call, there MUST NOT be a space between the method or function name and the opening parenthesis.
- Function-declaration: Spaces should be properly placed in a function declaration.
- Line-after-namespace: There MUST be one blank line after the namespace declaration. Lowercase-constants: The PHP constants true, false, and null MUST be in lower case.

Database Bottlenecks

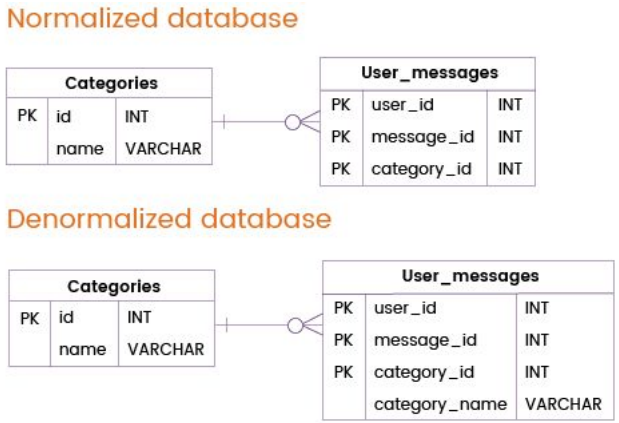
- Memory
  - find memory leaks
  - improve queries
  - add RAM
- Disk I/O use
  - replace existing disks with faster ones
  - reconsider usage of indexes
- CPU
  - lack of hardware resources
  - poor database design
  - non-optimal queries

Database optimization

- Partitioning
- SQL execution efficiency
- Order columns based on selectivity
- Finding a cost of an index
- Estimating workloads
- Sizing data
- Testing, debugging and validating a design

Partitioning

- Vertical Partitioning:
  - Improves access to data
  - Denormalize data



- Horizontal Partitioning:
  - Improves read/write performance
  - Distribute a table into more than one database servers

SQL execution efficiency

- Un-optimized query
  - Select *
  - From table
  - Where address LIKE ‘Heirnich Heine’

- Optimized query
  - Select *
  - From table
  - Where address LIKE :address
  - Set parameter :address, ‘Heinrich Heine’

Finding a cost of an index

- Indexes consume resources such as:
  - Disk space
  - CPU
  - I/O capacity
- Insert on row with 3 indexes is 10 slower than insert on row without any indexes
- For insert heavy applications we should reconsider using indexes

Sizing Data – Estimating Workloads

- Difficult to predict growing data
- Unused indexes (after deletions, insertions, updates)
- Try to rebuild indexes

- Ways to predict data
  - Extrapolating from a similar system
  - Benchmarking

Trip.me

- Image compression (lossless)
- JavaScript files to the bottom
- Move png images to CDN
- Cache images (expiration and validation)
- Minify JavaScript and CSS files (YUI compressor)
- Caching on responses on Symfony
- Usage of stored procedures in SQL queries
- Enabling on Zend Opcache

Results

- Decrease of average page load time from 6.5 to less than 5 seconds
- Decrease of time to first byte from 1 second to 0.7 seconds
- PageSpeed score: 81% => 91%
- Yslow score: 74% => 80%