

Meu site Plone está lento. O que fazer?

Fabiano Weimar dos Santos

xiru@xiru.org



Roteiro

- Por que o Plone é Lento?
- Performance Tuning
- Dicas e Truques

Por que o Plone é Lento?

- Fato: qualquer software com grande quantidade de acessos simultâneos costuma ser lento.
- Não importa a tecnologia adotada!
- O que faz a diferença: atenção com a **infra-estrutura** e com a **qualidade**.

Fatos

- Quanto maior o tráfego de um portal, mais especializada deverá ser a solução.
- Frameworks genéricos geralmente não tem boa performance.
- Frameworks agilizam o desenvolvimento, mas costumam impactar na performance.

Performance Tuning

- Mesmo que você considere algo rápido, efetue medidas que avaliem a performance em cenários com centenas de usuários simultâneos.
- O fenômeno das “redes sociais” pode, sem aviso prévio, ser um DOS.

Não Chute!
Adote métricas.

Ferramentas

- Medir performance de websites não é uma tarefa trivial.
- Ferramentas dificilmente simulam adequadamente o comportamento das aplicações “reais”.
- Há muitas ferramentas de Benchmark
 - <http://www.opensourcetesting.org/performance.php>

Ferramentas

- Apache Benchmark - ab
 - Acompanha o Apache HTTP Server
 - Costuma ser a forma mais simples de medir a performance de um “request”
 - Não avalia o carregamento de css, javascript, imagens, etc.
 - Não leva em consideração a eficiência do cache dos navegadores.

Ferramentas

Siege - HTTP load testing and
benchmarking utility

<http://www.joedog.org/index/siege-home>

Ferramentas

Flood - a profile-driven HTTP load tester

<http://httpd.apache.org/test/flood/>

É importante ter uma medida de quanto rápido é um site, com um determinado número de acessos simultâneos.

Dicas de Otimização de Performance

- Geralmente é difícil dar dicas “genéricas” de otimização de performance
- Em linhas gerais, quanto menos páginas o servidor processa, mais rápido é o site como um todo (cache)

Dicas de Otimização de Performance

- Nem todo site precisa de cache
- Há situações onde fazer cache significa apenas mais processamento e, de fato, queda de performance (Youtube)
- Desde que não exista processamento envolvido, nada é mais rápido do que servir conteúdo estático

Apache mod_rewrite

```
RewriteCond /PATH_WWW/DOMINIO/{REQUEST_FILENAME} -f  
RewriteCond %{REQUEST_FILENAME} !"  
RewriteCond %{HTTP_COOKIE} !__ac=  
RewriteCond %{HTTP:Authorization} !"  
RewriteCond %{HTTP:If-None-Match} !"  
RewriteRule ^/(.*) /DOMINIO/$1 [L]
```

```
RewriteCond /PATH_WWW/DOMINIO/{REQUEST_FILENAME}/index.html -f  
RewriteCond %{REQUEST_FILENAME} !"  
RewriteCond %{HTTP_COOKIE} !__ac=  
RewriteCond %{HTTP:Authorization} !"  
RewriteCond %{HTTP:If-None-Match} !"  
RewriteRule ^/(.*) /DOMINIO/$1 [L]
```

Apache mod_rewrite

- `$ cd PATH_WWW`
- `$ wget -m -np http://DOMINIO`

Cache Compartilhado

- Nem sempre servir apenas conteúdo estático é viável
- CMS geralmente não se preocupam em fazer “static deploy”; apenas publicam páginas dinâmicas
- A solução mais simples costuma ser adotar uma camada de webcache (Squid, Varnish, etc)

Cache Compartilhado

- Use o proxy para barrar tráfego indesejado
 - Bots de Indexação (Googlebot, msnbot, Yahoo Slurp, etc) podem significar até 35% do tráfego
 - Tráfego de bots é bastante custoso, pois não tem caráter repetitivo, desconsidera a relevância e visita o conteúdo “em profundidade”

Bloqueio de Bots (Squid)

```
acl badrobot browser -i Twiceler
```

```
acl badrobot browser -i Yeti
```

```
acl badrobot browser -i Daumoa
```

```
http_access deny badrobot
```

```
acl bot browser -i bot
```

```
acl bot browser -i crawler
```

```
acl bot browser -i Slurp
```

```
acl horario_comercial time MTWHF 06:00-23:00
```

```
http_access deny horario_comercial bot
```

Cache Compartilhado

- Para que o cache compartilhado de páginas dinâmicas seja efetivo é importante que as páginas sejam geradas com cabeçalhos HTTP adequados
- Há extensões do Firefox que permitem a análise facilitada de “headers” HTTP, como a “web developer,” firebug, YSlow (minha predileta)

Mapas Notícias Grupos Livros Gmail mais xirumacanudo@gmail.com | Página inicial clássica | Minha conta

iGoogle

Pesquisa Google Estou com sorte

Pesquisar na Web Pesquisar páginas em Português

Adicionar uma guia Alterar o tema (Earth-light) Adicionar novidades

Console HTML CSS Script DOM Net YSlow

Components | Statistics | Tools Rulesets Classic(V1) Edit Printable View Help

Speed up your web pages with YSlow

YSlow gives you:

- Grade based on the performance of the page (you can define your own ruleset)
- Summary of the page components
- Chart with statistics
- Tools for analyzing performance, including Smush.it™ and JSLint

Autorun YSlow each time a web page is loaded

Run Test

[» Learn more about YSlow and the Yahoo! Developer Network](#)

© 2009 Yahoo! Inc. All rights reserved.

YSlow

YSlow

Console HTML CSS Script DOM Net YSlow

Components | **Statistics** | Tools Rulesets YSlow(V2) Edit Printable View Help

Overall performance score 87 Ruleset applied: YSlow(V2) URL: http://www.google.com/ig

FILTER BY: CONTENT (6) | COOKIE (2) | CSS (6) | IMAGES (2) | JAVASCRIPT (4) | SERVER (5)

fewer HTTP requests

Grade A on Make fewer HTTP requests

This page has 9 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 © 2009 Yahoo! Inc. All rights reserved.

- Content Delivery Network (CDN)
- Expires headers
- Press components with gzip
- JS at top
- JavaScript at bottom
- CSS expressions
- JavaScript and CSS external
- Reduce DNS lookups
- JavaScript and CSS
- URL redirects
- Remove duplicate JavaScript and CSS

Fewer HTTP requests

YSlow

Grade **B** Overall performance score 87 Ruleset applied: YSlow(V2) URL: http://www.google.com/ig

ALL (22) FILTER BY: CONTENT (6) | COOKIE (2) | CSS (6) | IMAGES (2) | JAVASCRIPT (4) | SERVER (5)

A	Make fewer HTTP requests
F	Use a Content Delivery Network (CDN)
D	Add Expires headers
A	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
A	Reduce DNS lookups
A	Minify JavaScript and CSS
A	Avoid URL redirects
A	Remove duplicate JavaScript and CSS
A	Configure entity tags (ETags)

Grade D on Add Expires headers

There are 3 static components without a far-future expiration date.

- (2009/10/1) http://skins.gmodules.com/ig/skin_xml_to_css?...
- (2009/9/30) <http://br.advfn.com/p.php?...>
- (2009/9/30) <http://br.advfn.com/p.php?...>

Web pages are becoming increasingly complex with more scripts, style sheets, images, and Flash. On a first-time visit to a page may require several HTTP requests to load all the components. By using Expires headers these components become cacheable, which avoids unnecessary HTTP requests on subsequent views. Expires headers are most often associated with images, but they can and should be used on other components including scripts, style sheets, and Flash.

[»Read More](#)

Add Expires headers

YSlow tool interface showing performance analysis for <http://www.google.com/ig>. The overall performance score is 87, resulting in a Grade B. The ruleset applied is YSlow(V2).

Grade **B** Overall performance score 87 Ruleset applied: YSlow(V2) URL: <http://www.google.com/ig>

ALL (22) FILTER BY: [CONTENT \(6\)](#) | [COOKIE \(2\)](#) | [CSS \(6\)](#) | [IMAGES \(2\)](#) | [JAVASCRIPT \(4\)](#) | [SERVER \(5\)](#)

A	Make fewer HTTP requests
F	Use a Content Delivery Network (CDN)
D	Add Expires headers
A	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
A	Reduce DNS lookups
A	Minify JavaScript and CSS
A	Avoid URL redirects
A	Remove duplicate JavaScript and CSS

Grade F on Use a Content Delivery Network (CDN)

There are 15 static components that are not on CDN.

- http://skins.gmodules.com/ig/skin_xml_to_css?...
- http://www.google.com/ig/extern_js/f/CgJlbhICdXMrMMkBOAUsKzDtATgCLA/AUsKeuSLWP...
- http://skins.gmodules.com/ig/skin_fetch?...
- http://skins.gmodules.com/ig/skin_fetch?...
- <http://skins.gmodules.com/ig/images/logos/approved/white.png>
- http://skins.gmodules.com/ig/skin_fetch?...
- http://skins.gmodules.com/ig/skin_fetch?...
- http://skins.gmodules.com/ig/skin_fetch?...
- http://skins.gmodules.com/ig/skin_fetch?...
- <http://img0.gmodules.com/ig/images/cleardot.gif>

Use a Content Delivery Network (CDN)

Console HTML CSS Script DOM Net YSlow

Grade Components Statistics Tools

Rulesets YSlow(V2) Edit

Printable View Help

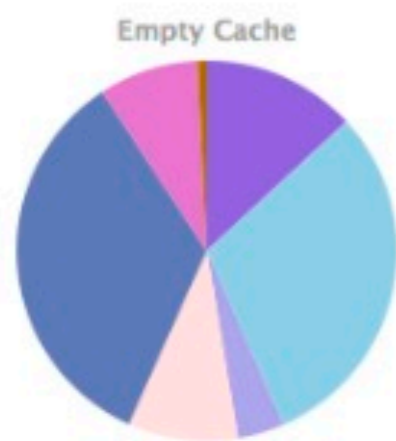
Components The page has a total of **20** components and a total weight of **144.3K** bytes

▲ TYPE	SIZE (KB)	GZIP (KB)	COOKIE RECEIVED (bytes)	COOKIE SENT (bytes)	HEADERS	URL	EXPIRES (Y/M/D)	RESPONSE TIME (ms)
[-] doc (1)	56.8K							
doc	56.8K	18.9K			🔍	http://www.google.com/ig	2009/9/30	27
[-] js (2)	134.7K							
js	82.4K	26.2K			🔍	http://www.google.com/ig/extern_js/f/CgJlbhICdXMrMMkBOAUsKzDtATgCLA/AUsKeuSLWpk.js	2010/9/28	31
js	52.3K	17.4K			🔍	http://www.ig.gmodules.com/gadgets/js/core:core.iglegacy:core.io.js?...	2010/9/28	79
[-] css (1)	22.9K							
css	22.9K	5.7K			🔍	http://skins.gmodules.com/ig/skin_xml_to_css?...	2009/10/1	28
[+] iframe (4)	36.2K							
[+] cssimage (9)	49.1K							
[+] image (2)	13.2K							

YSlow - Components

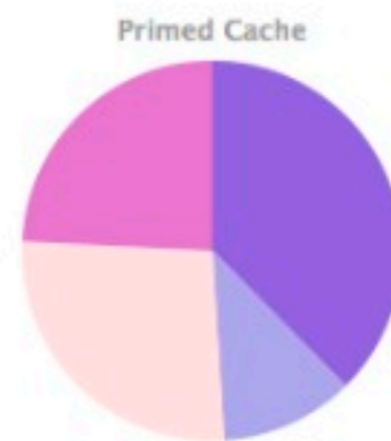
Statistics The page has a total of **20** HTTP requests and a total weight of **144.3K** bytes with empty cache

WEIGHT GRAPHS



HTTP Requests - 20
Total Weight - 144.3K

1 HTML/Text	18.9K
2 JavaScript File	43.7K
1 Stylesheet File	5.7K
4 IFrame	13.4K
9 CSS Image	49.1K
2 Image	12.2K
1 Favicon	1.1K



HTTP Requests - 8
Total Weight - 50.3K

1 HTML/Text	18.9K
1 Stylesheet File	5.7K
4 IFrame	13.4K
2 Image	12.2K

Copyright © 2009 Yahoo! Inc. All rights reserved.

YSlow - Statistics

Apache mod_expires

ExpiresActive On

ExpiresDefault "access plus 5 minutes"

ExpiresByType image/gif "access plus 1 day"

ExpiresByType image/jpeg "access plus 1 day"

ExpiresByType image/png "access plus 1 day"

ExpiresByType text/css "access plus 1 day"

ExpiresByType application/x-javascript "access plus 1 day"

Dicas de Otimização de Performance

- Cuidado com dependências remotas
- browser -> application server -> webservice -> sgbd -> storage...
- Num cluster, o maior custo costuma não ser apenas processador, mas sim a latência de rede

Dicas de Otimização de Performance

- Evite acordar muitos objetos no SGBD
- Evite conexões com SGBD
- Evite conexões LDAP
- Evite conexões HTTP (RSS, SOAP)
- Não faça conexões sem timeout

**Não tente adivinhar
porque seu site está
lento: use um profiler!**

Dicas de Otimização de Performance

- Use NTP para sincronizar relógios dos proxies, servidores de aplicação e banco de dados
- Evite gerar páginas dinâmicas com headers que expiram muito rápido
- Falta de sincronia de relógios pode acarretar a geração de headers “no passado”

Obrigado!

Fabiano Weimar dos Santos

xiru@xiru.org

