

# サーバサイドチューニングの基本

by 3100

時はWeb大戦争時代。

各社が1ミリ秒でも速くWebサイトを表示させるために  
しのぎを削る。

Google 「サイト表示が0.5秒遅くなると、検索数が20%減少する」

Amazon 「サイト表示が0.1秒遅くなると売上げが1%減少し、1秒高速化すると10%の売上が向上する」

3100 「サイト表示が1秒を超えると、そのサイトはTwitterで晒す」

UI研究で知られるヤコブ・ニールセン([Wikipedia](#))も Webの応答に1秒を超えると人が不快を感じ始めると述べた。

cf. [Website Response Times](#)

あなたのWebサービスは1秒以内で応答が完了しますか？



# 高速化の基本方針

- 通信サイズを減らす
  - Amazon S3や Google Cloud Storageの運用コスト削減にもつながる
- 通信回数を減らす
  - キャッシュを活用する




# 1. 測定ツールを知る


- GTmetrix

# GTmetrix

## GTmetrix | Website Speed and Performance Optimization

[Features](#)[Resources](#)[GTmetrix PRO](#)


Log InSign Up





### Latest Performance Report for:

<http://b.hatena.ne.jp/>

Report generated: Tue, May 31, 2016, 4:55 AM -0700




Test Server Region:  Vancouver, Canada

Using:  Firefox (Desktop) 45.0.2,  
PageSpeed 1.12.16, YSlow 3.1.8

 Looks like you might not be using a CDN  
[Why should I use a CDN? »](#)

Re-TestComparePage SettingsMonitorSet Up AlertsDownload PDF

Share This Report



### Performance Scores

PageSpeed Score	YSlow Score
<b>E (55%)</b> ▼	<b>D (63%)</b> ▼

### Page Details

Page Load Time	Total Page Size	Requests
<b>13.1s</b> ▼	<b>6.50MB</b> ▼	<b>344</b> ▼

[PageSpeed](#)[YSlow](#)[Waterfall](#)[Video](#)[History](#)

RECOMMENDATION	GRADE	TYPE	PRIORITY
▼ <b>Defer parsing of JavaScript</b>	F (0) ▼	JS	HIGH
▼ <b>Serve scaled images</b>	F (0) ▼	IMAGES	HIGH
▼ <b>Specify image dimensions</b>	F (0) ▼	IMAGES	HIGH

#### What do my scores mean?

**Rules are sorted in order of impact upon score**  
Optimizing rules at the top of the list can greatly improve your overall score.

# GTmetrix

基本的にこれさえ見れば十分。

- PageSpeed Insights、YSlow など複数の指標に対応。
  - 古い指標もあるので、100点を目指さないことも大事。
- 非ログインユーザでも計測結果を過去と見比べたりできる。
- 残念ながらmeta refreshやJSによるリダイレクトには非対応です。

# 指標1 : YSlow

YSlow – Official Open Source Project Website

- Yahoo! が提供しているWebサイトパフォーマンス計測ツール。
- 2015年5月で公式Twitterのつぶやきが最後となっている。
- 一部の指標が、今では古いと思われるもののまだまだ有用。

## 指標2 : PageSpeed Insights

### PageSpeed Insights

- Google が提供しているWebサイトパフォーマンス計測ツール。
- スマホ対応もしていて、今時感がある。
- 結果がかなりシンプル。人によっては良さそう。

## 2. 具体的な評価項目を試みる

- YSlow
- PageSpeed Insights (以下、PageSpeed)

## YSlowの評価項目 1/5

- Add Expires headers  
(キャッシュ期間を長くしよう)
- Use a CDN  
(静的ファイルをCDNに置く)
- Make fewer HTTP requests  
(1回のHTTPリクエスト数を減らす)
- Reduce DNS lookups  
(1回のリクエスト内にあるドメイン数を減らす)
- Compress components with gzip  
(テキストファイルはgzip化する)

## YSlowの評価項目 2/5

- Use cookie-free domains  
(静的リソースを別ドメインに置き通信サイズを減らす)
- Minify JavaScript and CSS  
(JS、CSSを最小化する)
- Avoid empty src or href  
(ブラウザによって無駄な処理を行う)
- Make AJAX cacheable  
(AJAXリクエストもキャッシュ化する)
- Put CSS at the top  
(CSSはheadタグに置く。metaタグの次くらいが良さそう)



## YSlowの評価項目 3/5

- Remove duplicate JavaScript and CSS  
(重複したJavaScriptやCSSを除く)
- Put JavaScript at bottom  
(JSは並行ダウンロードされないなので、body内下部に置き、後回しにする)
- Avoid AlphaImageLoader filter  
(IE系8以前のテクニックなので、気にしなくて良さそう。透過画像処理でメモリ消費量が増える)
- Avoid HTTP 404 (Not Found) error  
(各種リソースのリンク切れは無駄な通信となる。JS系でこれをやると最悪)

## YSlowの評価項目 4/5

- Reduce the number of DOM elements  
(大量のDOMがあると描画などが重くなる)
- Do not scale images in HTML  
(画像は必要なサイズで作成する。大きい画像を小さく表示するのは無駄)
- Use GET for AJAX requests  
(POSTはヘッダとデータを2回に分けて送信するので、GETの方が良い。IEは2KB制限注意)
- Avoid CSS expressions  
(IE独自拡張でJSをCSS内に記述できる機能。描画のたびに評価されて重たい原因となる)

## YSlowの評価項目 5/5

- Reduce cookie size  
(cookieはHTTPリクエストのたびに送信されてしまうので、なるべくサイズを小さく保つ)
- Make favicon small and cacheable  
(faviconは小さくし、キャッシュ可能にする)
- Configure entity tags (ETags)  
(ETagを付けるとファイルの変更を見てキャッシュを利用するか判断できるようになる。分散環境ではETagの生成を統一すること)
- Make JavaScript and CSS external  
(JSとCSSは外部ファイル化しよう。キャッシュ化と最小化の恩恵を受けられる)

## PageSpeedの評価項目（YSlowとの重複以外） 1/5

- Avoid a character set in the meta tag  
(IE8ではダウンロード周りで問題があった。代わりにHTTP Content-Typeレスポンスヘッダを使う)
- Specify a Vary: Accept-Encoding header  
(一部の公開proxyでは圧縮コンテンツに対応していないので、この設定によって圧縮/非圧縮両方のデータをproxyに格納させる)
- Specify image dimensions  
(画像サイズを明示することで無駄な再描画を防ぐ。CSSで指定しても可)

## PageSpeedの評価項目（YSlowとの重複以外） 2/5

- `Defer parsing of JavaScript`  
(JSの構文解析処理を必要になるまで先延ばしにする。[3つの例を紹介\(英語\)](#))
- `Remove query strings from static resources`  
(proxyサーバによってキャッシュ化されなくなるので、“?”付きURLをやめる)
- `Specify a character set early`  
(HTTPヘッダ `Content-Type`にて `charset`を指定し、ブラウザによる文字コード識別処理を不要にする)
- `Optimize images`  
(画像を圧縮する。Gtmetrixでロスレス圧縮後のファイルを勝手に作ってくれるのでよければ使う)

## PageSpeedの評価項目（YSlowとの重複以外） 3/5

- Minify HTML  
(HTMLを最小化する)
- Avoid landing page redirect  
(トップページでリダイレクトをしない。リダイレクトは体感速度の低下となる)
- Inline small CSS  
(小さいCSSファイルは埋め込むことでリクエスト数を減らせる。代わりに複数CSSファイルをまとめても良い)
- Inline small JavaScript  
(上と同様。)

## PageSpeedの評価項目（YSlowとの重複以外） 4/5

- Enable Keep-Alive  
(HTTPレスポンスヘッダでKeep-Aliveを有効にすると、同じTCPを使い回すので無効時よりTCP接続コストを減らせる)
- Minimize redirects  
(リダイレクトを減らすことでRTT=往復遅延時間を減らすことができる)
- Avoid CSS @import  
(@importを使うと並列ダウンロードができないので、なるべく使わない)

## PageSpeedの評価項目（YSlowとの重複以外） 5/5

- Combine images using CSS sprites  
(小さい画像を1ファイルにまとめてCSSスプライトを使うと、HTTPリクエストのオーバーヘッドを減らせる)
- Avoid document.write  
(document.writeによるパース、外部スクリプト読み込みなどは避ける)
- Avoid Flash on mobile webpages  
(Flashは多くのモバイル機器で対応していないので、HTML5で代替する)



### 3. 具体的に1つずつ改善していく

GTmetrixやYSlow、PageSpeedでは  
改善できそうな問題がある場合  
どのようにすればよいかを教えてください。

とはいえ

たとえばHTTPヘッダに関する知識などが必要なこともあります。  
とにかくググれば、先人たちの解を学ぶことができるでしょう。

焦らず、1つずつ理解して改善していきましょう。



## 参考リンク集

- [GTmetrix](#)による各項目の概説
- [PageSpeed Insights](#) でのモバイル解析 | [PageSpeed Insights](#) | [Google Developers](#)