Check your Site Performance score with Gtmatrix

GTMetrix is one of the tools that we have found ourselves using more and more to get detailed reports about our site's performance. It is a free tool that analyzes your page's speed performance using Google Page Speed and YSlow. GTMetrix then generates scores for your pages and offers actionable recommendations on how to fix them.

1).Leverage browser caching:

Reduce the load times of pages by storing commonly used files from your website on your visitors browser.

Reduce page load times for repeat visitors

Particularly effective on websites where users regularly re-visit the same areas of the website

Cost-benefit ratio: high

Access needed

What is browser caching?

Every time a browser loads a webpage it has to download all the web files to properly display the page. This includes all the HTML, CSS, javascript and images.

Some pages might only consist of a few files and be small in size - maybe a couple of kilobytes. For others however there may be a lot of files and these may add up to be several megabytes large. Twitter.com for example is 2mb+.

The issue is twofold.

These large files take longer to load and can be especially painful if you're on a slow internet connection (or a mobile device).

Each file makes a separate request to the server. The more requests your server gets simultaneously the more work it needs to do, only further reducing your page speed.

Browser caching can help by storing some of these files locally in the user's browser. Their first visit to your site will take the same time to load, however when that user revisits your website, refreshes the page, or even moves to a different page of your site, they already have some of the files they need locally.

This means the amount of data the user's browser has to download is less, and fewer requests need to be made to your server. The result? Decreased page load times.

How does it work?

Browser caching works by marking certain pages, or parts of pages, as being needed to be updated at different intervals. Your logo on your website, for instance, is unlikely to change from day to day. By caching this logo image, we can tell the user's browser to only download this image once a week. Every visit that user makes within a week would not require another download of the logo image.

By the webserver telling the browser to store these files and not download them when you come back saves your user's time and your web server bandwidth.

Why is it important?

The main reason why browser caching is important is because it reduces the load on your web server, which ultimately reducing the load time for your users.

How to leverage browser caching

To enable browser caching you need to edit your HTTP headers to set expiry dates on certain types of files.

Find your .htaccess file in the root of your domain, this file is a hidden file but should show up in FTP clients like FileZilla or CORE. You can edit the htaccess file with notepad or any form of basic text editor.

In this file we will set our caching parameters to tell the browser what types of files to cache.

```
## EXPIRES CACHING ##

<IfModule mod_expires.c>

ExpiresActive On

ExpiresByType image/jpg "access plus 1 year"

ExpiresByType image/jpeg "access plus 1 year"

ExpiresByType image/gif "access plus 1 year"

ExpiresByType image/png "access plus 1 year"

ExpiresByType text/css "access plus 1 month"

ExpiresByType application/pdf "access plus 1 month"

ExpiresByType text/x-javascript "access plus 1 month"

ExpiresByType application/x-shockwave-flash "access plus 1 month"

ExpiresByType image/x-icon "access plus 1 year"
```

ExpiresDefault "access plus 2 days"

</IfModule>

EXPIRES CACHING

Depending on your websites files you can set different expiry dates. If certain types of files are updated more frequently, you would set an earlier expiry date on the (ie. css files)

When you are done save the file as is and not as a .txt file.

If you are using any form of CMS, cache extensions or plugins might be available.

Recommendations

Be aggressive with your caching for all static resources

Expiry at a minimum of one month (recommended: access plus 1 year)

Don't set your caching more than a year in advance!

2). Combine image using CSS sprints:-

Combining images into as few files as possible using CSS sprites reduces the number of round-trips and delays in downloading other resources, reduces request overhead, and can reduce the total number of bytes downloaded by a web page.

CSS Example:

```
.home

{
    background: url(sprite.png ') 0 -45px;

}
```

```
//0 -45 indicates position for image.(left,right,top,bottom).
```

3). Minimize redirects:-

Minimizing HTTP redirects from one URL to another cuts out additional RTTs and wait time for users.

Sometimes it's necessary for your application to redirect the browser from one URL to another. There are several reasons web applications issue redirects:

- To indicate the new location of a resource that has moved.
- To track clicks and impressions and log referring pages.
- To reserve multiple domains, allow for "user-friendly" or "vanity" domains and URLs, and catch misspelled/mistyped URLs.
- To connect between different parts of a site or application, different country-code top-level domains, different protocols (HTTP to HTTPS), different security policies (e.g. unauthenticated and authenticated pages) etc.
- To add a trailing slash to URL directory names to make their contents accessible to the browser.

Whatever the reason, redirects trigger an additional HTTP request-response cycle and add round-trip-time latency. It's important to minimize the number of redirects issued by your application - especially for resources needed for starting up your homepage. The best way to do this is to restrict your use of redirects to only those cases where it's absolutely technically necessary, and to find other solutions where it's not.

4). Specify image dimensions:-

Page Speed currently only detects image dimensions that are specified via the image attributes. If you are specifying the dimensions via CSS, then you can safely ignore this recommendation.

Details from Google

When the browser lays out the page, it needs to be able to flow around replaceable elements such as images. It can begin to render a page even before images are downloaded, provided that it knows the dimensions to wrap non-replaceable elements around. If no dimensions are specified in the containing document, or if the dimensions specified don't match those of the actual images, the browser will require a reflow and repaint once the images are downloaded. To prevent reflows, specify the width and height of all images, either in the HTML <img> tag, or in CSS.

5). Minify CSS:-

Compacting CSS code can save many bytes of data and speed up downloading, parsing, and execution time.

Before:

```
/* cyrillic-ext */
@font-face {
  font-family: 'Open Sans';
  font-style: normal;
  font-weight: 300;
  src: local('Open Sans Light'), local('OpenSans-Light'),
url(http://fonts.gstatic.com/s/opensans/v10/DXI1ORHCpsQm3Vp6mXoaTSUUniRZcd_wq
8DYmIfsw2A.woff2) format('woff2');
  unicode-range: U+0460-052F, U+20B4, U+2DE0-2DFF, U+A640-A69F;
}
/* cyrillic */
@font-face {
  font-family: 'Open Sans';
  font-style: normal;
  font-weight: 300;
  src: local('Open Sans Light'), local('OpenSans-Light'),
url(http://fonts.gstatic.com/s/opensans/v10/DXI1ORHCpsQm3Vp6mXoaTeXREeHhJi4GE
UJI9ob_ak4.woff2) format('woff2');
  unicode-range: U+0400-045F, U+0490-0491, U+04B0-04B1, U+2116;
}
```

After:

```
@font-face{font-family:'Open Sans';font-style:normal;font-
weight:300;src:local('Open Sans Light'), local('OpenSans-Light'),
url(http://fonts.gstatic.com/s/opensans/v10/DXI1ORHCpsQm3Vp6mXoaTXhCUOGz7vYGh
680lGh-uXM.woff) format('woff');}
@font-face{font-family:'Open Sans';font-style:normal;font-
weight:400;src:local('Open Sans'), local('OpenSans'),
url(http://fonts.gstatic.com/s/opensans/v10/cJZKeOuBrn4kERxqtaUH3T8E0i7KZn-
EPnyo3HZu7kw.woff) format('woff');}
```

6). Minify JS:-

Same way as Minify css.

7). Optimize images:-

Reduce the load times of pages by loading appropriately sized images.

- Reduce file sizes based on where images will be displayed
- Resize image files themselves instead of via CSS

- Save files in appropriate format depending on usage
- Cost benefit ratio: high value

What is optimizing images for the web?

The images you create in programs like Photoshop and Illustrator look amazing but often the file sizes are very large. This is because the images are made in a format which makes them easier to manipulate in different ways.

With file sizes upwards of a couple of megabytes per image, if you put these files on your website it would be very slow to load.

Optimizing your images for the web means saving or compiling your images in a web friendly format depending on what the image contains.

Images hold data other than just the pixels we see on the screen. This data can add unnecessary size to the image which leads to longer load times as the user waits for the image to download.

In terms of cost versus benefit optimizing your images should be near the top of your page speed optimizations if you don't have them optimized already.

How does it work?

In simple terms optimizing your image works by removing all the unnecessary data that is saved within the image to reduce the file size of the image based on where it is being used in your website.

Optimizing images for the web can reduce your total page load size by up to 80%.

There are two forms of compression that we need to understand, Lossy and Lossless.

Images saved in a lossy format will look slightly different than the original image when uncompressed. Keep in mind that this is only visible at a very close look. Lossy compression is good for web, because images use small amount of memory, but can be sufficiently like the original image.

Images saved in lossless format retain all the information needed to produce the original image. For this reason these images carry a lot more data and in return are a much large file size.

We also can optimize images for the web by saving them as the appropriate dimensions. Resizing the image on the webpage itself using CSS is helpful but the issue is the web browser will still download the entire original file, then resize it and display it.

Can you imagine take a poster size image and using it as a thumbnail? The little 20px by 20px image would take as long to load as the original poster, when we could just be loading a 20px image the whole time.

Why is it important?

The main reason why it's so important to optimize your images is because 90% of most websites are graphics dependent and therefore there are a lot of image files. Leaving these images uncompressed and in the wrong format can drastically slow down your web page load times.

How to Optimize Your Images

Full optimization of images can be quite an art to perfect as there are such a wide variety of images you might be dealing with. Here are the most common ways to optimize your images for the web.

- Reduce the white space around images - some developers use whitespace for padding which is a big no no. Crop your images to remove any whitespace around the image and use CSS to provide padding.
- Use proper file formats. If you have icons, bullets or any graphics that don't have too many colors use a format such as GIF and save the file with lower amounts of colors. If you have more detailed graphics then use JPG file format to save your images and reduce the quality.
- Save your images in the proper dimensions. If you are having to use HTML or CSS to resize your images, stop right there. Save the image in the desired size to reduce the file size.

To resize your images you will have to use some form of program. For basic compression you can use a simple editing program such as GIMP. For more advanced optimization you will have to save specific files in Photoshop, Illustrator or Fireworks.

8). Avoid bad requests:-

Removing "broken links", or requests that result in 404/410 errors, avoids wasteful requests.