|  |
| --- |
| Ekwa |
| Website Optimization Research |
| How to boost your website performance |
|  |
| **By Chamara – Kickoff Team** |
| **12/4/2014** |

# Table of Contents

[What is website optimization and why we need it? 2](#_Toc405418341)

[How to measure your webpage speed? 2](#_Toc405418342)

[Page speed/performance measuring tools 2](#_Toc405418343)

[Google Page Speed 2](#_Toc405418344)

[Yslow 4](#_Toc405418345)

[What we must do to maintain our site optimization 6](#_Toc405418346)

[1. Avoid Landing Page Redirects 6](#_Toc405418347)

[What is responsive web design and how can we use it to our HTML conversions. 6](#_Toc405418348)

[2. Configure the Viewport 8](#_Toc405418349)

[3. Improve Server Response Time, Leverage Browser Caching, Add compression (Gzip) 8](#_Toc405418350)

[4. Minify and Combine Resources (CSS, and JavaScript), Optimize Images 10](#_Toc405418351)

[Grunt JS 10](#_Toc405418352)

[Setting Grunt JS with SASS 14](#_Toc405418353)

[Conclusion 17](#_Toc405418354)

[References 19](#_Toc405418355)

# What is website optimization and why we need it?

Here, we are focused on speed and performance of a particular website, not the SEO side. Fast and optimized pages lead to higher visitor engagement, retention, and conversions. Therefore, in this age, site optimization is a critical factor for websites.

# How to measure your webpage speed?

There are few tools to measure your page speed online. But here, we are mainly focused on two tools. [Google Page Speed](https://developers.google.com/speed/pagespeed/insights/) and [Yslow](http://yslow.org/). These two has their own rule sets for make the page more optimized and faster.

# Page speed/performance measuring tools

## Google Page Speed

This tool is created by the tech giant Google and they have developed some other APIs and tools for automated optimization tasks like image optimization, file compression etc. currently they support only for Ubuntu and Fedora. But we can use JavaScript tools to get the same results. We will get to that later in this documentation.

When your site’s URL is pasted on [Google Page Speed](https://developers.google.com/speed/pagespeed/insights/) website and hit the analyze button Google will go through the page and compare the site with their set of rules and show the optimization score of the site with a bit detailed explanation. This is very helpful to understand what we did wrong and how to make them right.



Figure Google Page Speed online Analyzer



Figure Google page Speed Optimization Score view.

## Yslow

Yslow is another good tool to check your page speed created by another big company in the IT industry, Yahoo. Not like Page Speed Yslow does not have an online Analyzer, but a browser plugin that can measure the speed of the webpage. Also there are other methods too like CMD, Node.js server, phantom JS etc. But we will stick to the browser plugin for now.

You have to go to the [Yslow home page](http://yslow.org/) and install Yslow on your desired browser. They support major desktop browsers and there is a way to install it on the mobile browsers also.



Figure Yslow availability

After the installation of the browser plugin of Yslow and you can run it for your website from your browser window. After running it, a score for the site will be given and there will be sections for that score which is helpful to understand the right and wrongs of the page.



Figure Yslow Score

# What we must do to maintain our site optimization

As Google page speed and Yslow, there is a list of rules we can easily do to up the site optimization.

## Avoid Landing Page Redirects

Don’t have more than one redirect from the given url to the final landing page. This is why we use responsive web design. Therefore, no redirects to the landing page like [www.m.example.com](http://www.m.example.com). Fast and optimal!

### What is responsive web design and how can we use it to our HTML conversions.

Responsive web design’s name speaks for itself and it responds to the screen width of a browser. So, we don’t have to create a separate site for mobiles and tablets. Usually when we create a website, we consider main three screen types.

* Desktop (769px and above)
* Tablet (481px to 768px)
* Mobile (480px and below)

So, to build a good responsive site we can use different kind of methods. There are front end frameworks like Bootstrap, Foundation, skeleton, Gumby etc. which consist of inbuilt grid system that help to easily build a responsive website. Besides, some of them pack a complete front end UI kit with the framework like buttons, form elements, sliders, responsive galleries etc. But, we couldn’t use those properly at the moment since the designs are not done according to a grid system. So, for now we will go deeply into Dreamweaver’s fluid grid layout method.

#### Adobe Dreamweaver’s Fluid Grid Layout

This design method was introduced in Adobe CS6 and above versions in Dreamweaver. In this, CSS file is divided to three screen resolutions such as Desktop, Tablet and Mobile. Predefined media queries get generated once we select this in Dreamweaver and we can code our CSS according to our standards.

Open Dreamweaver >> File >> New >> Fluid Grid Layout



Figure 5 Responsive site which is done using Fluid Grid layout method

#### Mobile Site Standards



Figure Example mobile view of a responsive website

1. Call button, Appointment request/contact us form button, Email Button and directions button should be on the top of the mobile site as fixed nav-bar for easy access.
2. Main menu should contain all the main links and bucket links. Sub links can be there separately in the site. But there should be only one responsive menu for user friendliness.
3. Must go with the desktop site theme and similar to its initial design.
4. Should check the site in IOS, Android and Windows mobiles etc. (with [Browserstack](http://www.browserstack.com/start)).
5. Should be optimized correctly.

## Configure the Viewport

A viewport controls how a webpage is displayed on a mobile device. Without a viewport, mobile devices will render the page at a typical desktop screen width, scaled to fit the screen. Setting a viewport gives control over the page's width and scaling on different devices.

<meta name=viewport content="width=device-width, initial-scale=1">

## Improve Server Response Time, Leverage Browser Caching, Add compression (Gzip)

To outcome above factors and get optimum optimization you could add a simple code snippet to the site’s htaccess file. Following is the code.

#############Start Gzip####################################

<IfModule mod\_deflate.c>
 AddOutputFilterByType DEFLATE text/html text/plain text/css application/json
 AddOutputFilterByType DEFLATE text/css
 AddOutputFilterByType DEFLATE application/javascript
 AddOutputFilterByType DEFLATE text/xml application/xml text/x-component
 AddOutputFilterByType DEFLATE application/xhtml+xml application/rss+xml application/atom+xml
 AddOutputFilterByType DEFLATE image/x-icon image/svg+xml application/vnd.ms-fontobject application/x-font-ttf font/opentype
</IfModule>

#############End Gzip####################################

############## Start 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>

<FilesMatch "\.(jpg|jpeg|png|gif|js|css|ico|swf)$">
 Header set Expires "access plus 1 year"
</FilesMatch>

# 2 DAYS

<FilesMatch "\.(xml|txt|woff)$">
 Header set Expires "access plus 2 days"
</FilesMatch>

# 2 HOURS

<FilesMatch "\.(html|htm|php)$">
 Header set Expires "access plus 2 hours"
</FilesMatch>

############## End Expires Caching #########################

## Minify and Combine Resources (CSS, and JavaScript), Optimize Images

For these two tasks we can use JavaScript tool [Grunt JS](http://gruntjs.com/) which is really helpful and automate the repetitive work. Speaking of minifying CSS and JS it is very helpful to optimize the site speed and performance. The entire minified and combined css file should be header and all the scripts should be on the footer area before the end of the body close tag.

But when the CSS file and JS files are minified it is very hard to do edits and keep track of the code as the minified version is not easy for the eyes. Therefore, we need to find a way to minify the resources and at the same time easy to do edits and stuff. This is the time Grunt JS comes and help us to do both. So, let’s have some idea about Grunt JS and see how to install it and use it.

### Grunt JS

Grunt is basically a system for automating repetitive jobs.

For example, every time you save your Sass or JavaScript files, you will want certain actions to automatically run. For Sass files, you will want the Sass to compile down to CSS, and for JavaScript files you will want them to be automatically checked and unit tested.

Using Grunt you can have all the automations rather than using several tools out there.

#### Installing Grunt

Before you install Grunt you have to have Node.js installed since Grunt is a Node.js package. You can install Node.js from [here](http://nodejs.org/download/).

npm install -g grunt-cli

After Node.js installation just run below command on the command prompt

#### The structure of a Grunt project

A Grunt project basically consists of two main files. Firstly there is **package.json** and secondly there is **GruntFile.js**.

The package.json is essentially the same as your composer.json file in that it lists meta data about your project and what dependencies are required.

The GruntFile.js is used to configure, and define the tasks that you want to be run on your project. The GruntFile is basically just a valid Javascript file that has a set of configurations and tasks for building your project.

#### Setting up a new Grunt project

To set up a new Grunt project, we basically just need to create the **package.json** and **GruntFile.js** files.

#### Creating the package.json file

Firstly you could run npm init from Terminal. This will ask you a couple of questions about your project and then it will automatically create the package.json file for you.

Alternatively, you can use one of the Grunt init templates. grunt-init is a scaffolding tool for creating new projects from a set template.

To install grunt-init, simply run npm install -g grunt-init from Command Prompt.

Next you can create a new project from an existing template. See the [Grunt documentation](http://gruntjs.com/project-scaffolding) for full details of these templates.

#### Sample Package.js template

{

 "name": "dr-bengelsdorf",

 "version": "0.0.0",

 "description": "Website for Dr. Begelsdorf",

 "main": "Gruntfile.js",

 "dependencies": {

 "grunt": "^0.4.5",

 "grunt-contrib-jshint": "^0.10.0",

 "grunt-contrib-nodeunit": "^0.4.1",

 "grunt-contrib-compass": "^0.9.1",

 "grunt-contrib-watch": "^0.6.1",

 "grunt-contrib-uglify": "^0.5.1",

 "grunt-newer": "^0.7.0",

 "grunt-smushit": "^1.3.0"

 },

 "devDependencies": {

 "grunt-contrib-imagemin": "^0.9.1"

 },

 "scripts": {

 "test": "echo \"Error: no test specified\" && exit 1"

 },

 "author": "Ekwa Marketing",

 "license": "ISC"

}

Here I have used four useful packages for the above project. (Highlighted in green)

* **JShint** – this will help to find any javascript errors on the project
* **Compass** – this is a css framework that helps to compile SCSS to CSS and other CSS work.
* **Watch** – helps to push changes live into the result.
* **Uglify** – this helps to combine all the JS file to a one file and minify that file.

You can install packages via below command in CMD. You have to know the correct package names. Try Google.

npm install <package name> --save-dev

For instant, to install uglify package you need to run

npm install grunt-contrib-uglify --save-dev

#### Creating the GruntFile.js file

This is where all the real heavy lifting occurs when using Grunt, so you will want to get accustomed to it so you can tweak your project to run exactly how you want it to.

As I mentioned above, the GruntFile.js is a valid Javascript file, so if you are familiar with Javascript, this shouldn’t be too difficult to pick up.

//Wrapper

module.exports = function(grunt) {

 // Project configuration.

 grunt.initConfig({

 pkg: grunt.file.readJSON('package.json'),

 uglify: {

 build: {

 src: [

 'js/libs/jquery-1.11.1.min.js',

 'js/libs/jquery.flexslider.js',

 'js/libs/jquery.mmenu.min.all.js',

 'js/common.js'

 ],

 dest: 'js/build/global.min.js'

 }

 },

 //Other packages

 });

 //Load the plugin that provides the "uglify" task.

 grunt.loadNpmTasks('grunt-contrib-uglify');

 // Default task(s).

 grunt.registerTask('default', ['uglify']);

};

Above is the sample Gruntfile.js and there is only one package is installed, Uglify. We can have multiple packages in this file and after adding those we have to load the plugin and register the task.

Please follow up the [Grunt documentation](http://gruntjs.com/getting-started) to have a better idea.

### Setting Grunt JS with SASS



Figure Grunt JS with SASS

#### What is SASS?

Sass is a scripting language that is interpreted into Cascading Style Sheets (CSS). SassScript is the scripting language itself. Sass consists of two syntaxes. The original syntax, called "the indented syntax", uses syntax similar to Haml. It uses indentation to separate code blocks and newline characters to separate rules. **The newer syntax, "SCSS", uses block formatting like that of CSS**. It uses braces to denote code blocks and semicolons to separate lines within a block. The indented syntax and SCSS files are traditionally given the extensions .sass and .scss respectively.

**We use .SCSS format because its code is like CSS and easy to write.**

#### How to install SASS?

We will be using Compass (open source) to install sass and will be using it to create our .SCSS files.

#### What will Sass and Compass do?

With Sass and Compass you can generate optimized css output to use on our web projects. It will take your .sass or .scss source files and compile browser readable CSS for you.

Since Compass helps a lot with handling other assets than just CSS, we’ll be using it to generate our CSS.

#### How does Compass work?

Most projects I’ve worked on have their images, CSS and other assets separated in different folders. Compass will help us organize our .sass or .scss code to work better with our other assets.

#### The difference between .sass and .scss

Only thing different about the two is the syntax. Both will be parsed by Compass. In our projects we’ll use .scss. A more in-depth article of which syntax to pick can be found here: <http://thesassway.com/articles/sass-vs-scss-which-syntax-is-better>.

#### Install Compass on your computer

* First we will need to install Ruby on the computer because compass is running on Ruby.
* To install ruby follow this link. <http://rubyinstaller.org/>
* You just have to download it and install.
* Now we can install Compass. So let’s open up a Command Prompt.
* gem install compass (type it on the CMD window.)
* Now you are all done. You have successfully installed compass with SASS.

**Video Tutorial -** <https://www.youtube.com/watch?v=PRJpf-yyWZ4>

Now that you have successfully set up SASS with compass we can use it with Grunt for automated compilation of SCSS o CSS. Just run grunt-contrib-compass in CMD on your project root folder. Then add below code to the Grunfile.js within

module.exports = function(grunt) {

 grunt.initConfig({

 pkg: grunt.file.readJSON('package.json'),

 //Compass package goes here.

});

}

Full code is as below

module.exports = function(grunt) {

 grunt.initConfig({

 pkg: grunt.file.readJSON('package.json'),

 compass: {

 dist: {

 options: {

 sassDir: 'sass',

 cssDir: 'css'

 }

 }

 },

 watch: {

 css: {

 files: '\*\*/\*.scss',

 tasks: ['compass']

 }

 }

 });

 grunt.loadNpmTasks('grunt-contrib-compass');

 grunt.loadNpmTasks('grunt-contrib-watch');

 grunt.registerTask('default',['watch']);

}

Here we have used watch plugin for constantly compile the SCSS to CSS live. No need to run grunt command to compile. Just run grunt watch on CMD and changes will take effect on live.

#### Image Minify and compression

This is also a huge factor when it comes to website optimization. Luckily, we can do that with Grunt.js very easily. You can install it with below command on CMD on your project folder root.

You can learn how to use this on your Gruntfile.js from [here](https://github.com/gruntjs/grunt-contrib-imagemin).

# Conclusion

npm install grunt-contrib-imagemin --save-dev

Site optimization is a critical factor in performance of a website. The more it is optimized, the more it will perform. In this technological age, we cannot rely only on desktops. Users tend to use mobile devices to browse the web and websites should perform every platform. So, as webmasters we should always keep in touch with the new web technologies and use them for our day to day tasks to build performance, ease up the tasks and efficiency.

If you follow above things, you can build a good project with good performance at the end of the day. Don’t forget to evolve and add your own performance boosting tweaks to your projects and have a better output.

# References

* Google page speed insight rules - <https://developers.google.com/speed/docs/insights/rules>
* Yslow - <http://yslow.org/>
* SASS documentation - <http://sass-lang.com/documentation/file.SASS_REFERENCE.html>
* SASS basics - <http://sass-lang.com/guide>
* Grunt documentation - <http://gruntjs.com/getting-started>
* Grunt on Github with all plugins - <https://github.com/gruntjs>
* Node.js download - <http://nodejs.org/download/>
* Ruby download - <http://rubyinstaller.org/>
* Grunt basic learning video tutorial - <https://www.youtube.com/watch?v=Mr2VBRdRYak>
* Test server URL with site optimization help guides - [www.ekwadesign.com/kick-off-team/site-optimization/](http://www.ekwadesign.com/kick-off-team/site-optimization/)