# Magento for Designers: Optimization

Magento is a stunningly powerful e-commerce platform. In this miniseries, we’ll learn how to get started with the platform, getting to know the terminologies, setting up a store and all related aspects of it and finally learn how to customize it to make it our very own.

In this final part, we’ll be tackling one of the biggest complaints people have with Magento: its performance. We’ll review a number of different methods you could apply to significantly boost the platform. Excited? Let’s get started!

## The Full Series

* [Part 1: Installation and Setup](http://net.tutsplus.com/tutorials/php/magento-for-designers-part-i/)
* [Part 2: Products, Taxes, Categories, Payment Gateways, etc.](http://net.tutsplus.com/tutorials/php/magento-for-designers-part-2/)
* [Part 3: Theming](http://net.tutsplus.com/tutorials/php/magento-for-designers-part-3/)
* [Part 4: Building the Theme](http://net.tutsplus.com/tutorials/php/magento-for-designers-part-4/)
* [Part 5: Building the Product Information Page](http://net.tutsplus.com/tutorials/php/magento-for-designers-part-5/)
* [Part 6: Building the Home Page, with a Content Slider](http://net.tutsplus.com/tutorials/php/magento-for-designers-part-6/)
* [Part 7: The Fabled Multi-Store Functionality](http://net.tutsplus.com/tutorials/php/magento-for-designers-multi-store-functionality/)
* [Part 8: Optimization](http://net.tutsplus.com/tutorials/php/magento-for-designers-optimization/)

## A Quick Recap

In the last few parts, we took a high level look at how Magento themes are put together, the components that drive it and how everything works together to create a working theme along with a cursory look at a number of API methods that Magento exposes. We also took a focussed, step by step look at what Magento multi store is, how it can help you and finally how to set it all up.

## What Are We Going to do Today?

Magento’s performance is one area about which users complain consistently. Not a lot of it is true but there are a myriad different ways to improve the performance, of course. Today, we’ll be taking a look at ways to drastically improve the performance of your Magento installation. I know it sounds interesting and you’re itching to get started.

Quick note before we start though. The majority of the tweaks and/or optimizations mentioned here will be just those: mentions. You should probably google around for the implementation since the implementation itself will require a reasonably sized tutorial in itself. With that out of the way, let’s get started!

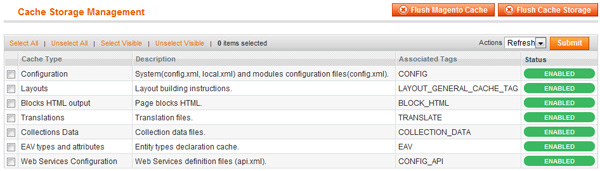
## Choose the Right Platform

This is the most critical aspect of your Magento installation. Magento can be run on a shared host but beware, the performance is not going to be optimal. Magento really requires more power than a run of the mill shared host can provide and it’ll really show when you make it run on one.

A VPS or a dedicated server is most ideal when it comes to Magento. And remember not to skimp on the memory because you’ll be needing it. Lots of it.

## Enable Caching

Letting Magento cache some parts of its code base is one of the few things that you should do when starting to optimize an installation. You need not enable caching for everything but I’d suggest layouts and block HTML output as essentials. With this caching enabled, depending on the size of the site, you can expect your load time to be decreased drastically.



Do keep in mind that you’ll need to disable caching whilst development. If you’re making changes to your code and it fails to reflect in the front end, caching is the culprit. Such an obvious point but it has been the cause of so much hair pulling during development.

## Turbocharge Your Cache

Once you’ve enabled caching, you can now take it one step further. Magento caches information using a file based storage system. While this is the most reliable way, there are often times when you want sheer speed. In these cases, you should look into deploying a memory based caching system.

There are a number of options available and people have had great success with them. eAccelerator is my tool of choice these days. Just remember to configure it with sufficient memory or it may have to purge data during the most inopportune moments.

## php.ini Tweaks

If you’re on a VPS or a dedicated server, look into disabling non-essential PHP extensions. Magento only needs a handful like PDO\_MySQL so feel free to disable the extraneous ones.

Additionally, there are 2 tweaks that will result in better performance.

By default, PHP is configured to use an absurdly low amount of memory. You’ll need to increase this. You can do this by adding the following line to the config file:

1. php\_value memory\_limit 512M

The second is by tweaking the realpath\_cache value to so.

1. realpath\_cache\_size=1M
2. realpath\_cache\_ttl=86400

## Apache Module Tweaks

This is more of a general topic than a Magento specific one so I’ll be brief here. There are a number of Apache modules that you’ll need to enable and/or setup to improve the performance of your store.

The first is mod\_deflate. This allows the output from the server to be drastically compressed, thus leading to better performance. Definitely a module that needs to be enabled.

Secondly, look into mod\_expires. This module allows the browser to cache specific resources for a set period of time so returning visitors will have a more snappier experience.

And finally, KeepAlives. This allows for persistent connections by creating long lived sessions allowing multiple requests to be routed through the same TCP connection. This helps when you have a number of small assets in your page.

## Rebuild Your Indexes

Remember to rebuild your indexes for the tables through the back end periodically. This is an essential point that will result in a noticeable performance boost to your store.

Do keep in mind that rebuilding the indexes is a very expensive affair, resource wise and should be done during off hours. Doing this during even moderate activity on your store will significantly degrade the store’s performance and responsiveness.

## Add a PHP Accelerator

The PHP interpreter has to read each file from disk and compile it on every request, something that’s extremely CPU hungry if you consider the sheer size of Magento. In these case, a PHP accelerator like APC or eAccelerator is highly recommended.

Tools like these increase the performance of your PHP code significantly by caching your code in intermediate byte code and is served instead of having to ask the interpreter to compile it every time.

## In-Memory Partition to Reduce IO Cycles

This is an extension of the point noted slightly above. Magento stores a lot of informations [not just the DB] in the server’s hard drive which as you know is the slowest link in the chain. This includes caches, session information and so on. If you want to improve performance in this area, the best way to move forward is to implement a memory based file system to make sure all that data is stored in-memory instead of costly calls to the hard drive.

Since we don’t have the space here, I’ll just point you in the right direction. What you’ll need to do is set up a tmpfs partition that takes care of your session and cache directories. These can be found under /var/www/domain.com/var/sessions/ and /var/www/domain.com/var/cache/ respectively.

## Enable Flat Front end

The flat front end catalog module creates additional database tables to store data in a flat/linear format. This module is available for both category and product data so feel free to enable it independently as required by your use case.

Enabling this module is only recommended for big stores. Stores with a small catalog should definitely stay away since enabling this may not provide enough of a performance surge to warrant usage and may actually decrease performance. Make a carefully considered decision here.

## Separate the Back End

Magento’s back end is significantly more resource intensive than the front end since it has to handle a lot more information. And once your store has reached a certain threshold with respect to the number of products, say, 5,000, it really starts to tax the server.

Having a dedicated admin server can help make the admin panel get the CPU cycles it needs with adversely affecting the performance of the rest of the site.

## Tweak Block Usage

The default Magento theme is a great place to start from but isn’t really the most efficient. Let me elaborate. The default theme is designed to exhibit all of Magento’s features and thus, naturally, bundles every whiz bang feature. Remember that Magento builds a page through parsing the layout we specify and thus the page generation time is directly related to the number of block you use in a page.

So when building your own theme, avoid needlessly complex layouts, simplify and/or combine the smaller blocks to reduce the CPU cycles it takes to render the page’s code.

## Add CSS/JS Intelligently

Magento has an elegant system in place to programmatically include/exclude style sheets and JavaScript files within themes. A lot of theme authors seem to completely bypass this system by just including these elements manually in the HTML templating. This is wrong.

The right way to include these assets is shown below:

[view plaincopy to clipboardprint?](http://net.tutsplus.com/tutorials/php/magento-for-designers-optimization/)

1. <reference name="head">
2. <action method="addJs"><script>mocha.js</script></action>
3. <action method="addJs"><script>nitro.js</script></action>
4. </reference>

By using the code above in the layout files, Magento combine the JS files into a single file and served to the browser in a single request.

## Serve Static Content Efficiently

The rules of optimizing normal sites or applications apply here as well. Browsers typically limit the number of downloads from a domain to 2-5 which is very limiting when you’re loading a media rich theme with a number of small assets. There are a number of easy ways out of this

The first and easiest is something you’re all familiar with: sprites. Using sprites, you can minimize the number of concurrent requests generated by your theme.

This approach may not work in all cases. What if your theme has a bunch of nifty scripts, videos and whatnots? These shouldn’t be block the loading of the rest of the page. In these cases, just create a sub-domain for static assets and point your Magento installation to the resource.



## Balance Your Logging

Logging is essential but is also costly since it’ll need to write to the disk. Magento, in general, is very chatty when it comes to logging and soon you’ll be knee deep in log files most of which you won’t need.

The solution here is to disable all but the essential parts. Have you enabled slow queries to be logged on your production server? Turn it off, right now! Slow queries must be investigated and fixed well before deployment. It has its place in a development server, not a production server.

## Keep Everything Updated

This is another essential tip that a lot of people keep forgetting. Whether it be the version of PHP, mySQL or Magento itself, the newer versions bring substantial speed and stability which you can’t quite afford to miss out.

Specially in the case of Magento, even each mini update fixes a number of performance issues and bug fixes so as long as the branch is labelled stable, upgrade!

## Magento Compiler

Magento has a spiffy [compiler module](http://www.magentocommerce.com/magento-connect/Magento+Core/extension/1359/magento-compiler) that promises anywhere between a 25%-50% speedup. As the name suggests, this module compiles all Magento files and thus leads to better performance.

Note that this module is still in beta and only supports 1.3x builds. This should be ready soon so keep this point in mind as well.

## Use Fooman Speedster

[Fooman Speedster](http://www.magentocommerce.com/magento-connect/FOOMAN/extension/457/fooman-speedster) is a powerful extension that can give your store a nice boost in performance. It works by combining, minifying, compressing and caching your JavaScript and CSS files. On top of this, it also enables GZIP compression for the rest of the store.

All in all, it means lesser data is sent and less requests which is a net win any way you see it. It works with all major version of Magento and is definitely a must have for any store.

## Optimize your DB

Another of those things that you should do periodically to improve performance. You’ll need to repair your database, if necessary and then optimize it. You can do it through your hosts CPanel, phpMyAdmin or just shell commands. [Here](http://www.webkeydesign.com/26/optimize-your-mysql-databases/) is a quick link walking you through the process using phpMyAdmin.

## The Last Word

And we are done! Today, we took a look at a number of ways to improve Magento’s performance. Hopefully this has been useful to you and you found it interesting. I’m sure I’ve missed a number of points here so please feel free to chime in the comments with the tips you’ve been using.

Questions? Nice things to say? Criticisms? Hit the comments section and leave me a comment. Happy