

## INFO

Website Performance Guidelines

Document Version: 1.0

Date: 2009.09.30

---

### Purpose

ExpressionEngine is an extremely versatile and powerful piece of software that is primarily used for web publishing. Because of its complexity, there are often numerous paths to building even the simplest page, and it is not always obvious which solution will be the best for performance for any given site. This document aims to pull back the curtain on the inner workings and nuances of site development in ExpressionEngine. By reading and understanding this document, you will gain insight into bringing the best possible performance to your ExpressionEngine-based site.

### Using Your Team

If you are in charge of a high profile and high trafficked site, you need to have good, constant relations with your team of designers, developers, and server administrators. They need to be informed of changes and problems, while you need to know what *they* know about how the site and its infrastructure are built and supported. Information and collaboration is key to keeping a heavily trafficked site running smoothly.

If your designers and developers are not familiar with ExpressionEngine, provide them with the resources to become so. There are numerous tutorials, courses, examples, and documentation for them to partake of in the ExpressionEngine community. Definitely give them this document to read. Suggest that if they have questions or concerns, they ask them in the ExpressionEngine forums. One of the greatest benefits of using ExpressionEngine is the top notch support from EllisLab.

If your site seems sluggish, contact your server administrator and ask if they have seen any large traffic spikes or if the MySQL Slow Query Log indicates that any queries are taking an exceptionally long time to process. If your server administrator is reluctant to help or is unable to provide more information, then you need to switch hosts. The backbone of your site is important, you should not be kept alone and in the dark.

## INFO

Website Performance Guidelines

Document Version: 1.0

Date: 2009.09.30

### Maintenance

Maintain your site and your software. ExpressionEngine, and the technologies it relies on, have many abilities to maintain your data and keep your site running smoothly. However, you still need to periodically perform maintenance and software upgrades to insure you are receiving the best performance. Here are a few to do items that you should stay on top of:

**Repair and Optimize Tables.** In the *Admin => SQL Manager* of ExpressionEngine's control panel, you can view all of your EE related database tables. These database tables contain all of the data for your site. By repairing and optimizing your tables with the pulldown at the bottom of the list of tables, you help MySQL fix any data problems, reclaim unused space, and defragment its data files.

**Remove old, unused EE Data.** Remove cruft that complicates your site and makes debugging complicated and may confuse new developers.

- **Templates.** By removing old Templates, you keep your design in good order and structure.
- **Weblogs. Custom Fields. Category Groups. Entries.** Remove old test data and content no longer used.
- **Members.** Periodically remove members who never moved beyond Pending status.
- **Add-Ons.** Not using them? Don't intend to use them? Remove.

**Upgrade ExpressionEngine.** ExpressionEngine is updated on a regular basis with new features, bug fixes, and various improvements. Keeping up to date will insure you are using the best, securest, and fastest version of ExpressionEngine available. Keep track of ExpressionEngine updates by subscribing to the ExpressionEngine blog (<http://expressionengine.com/blog/>) and the Builds forum (<http://expressionengine.com/forums/viewforum/38/>). Always follow the upgrade instructions in the ExpressionEngine documentation.

**Upgrade Add-Ons.** Just like ExpressionEngine, many of the Add-Ons used to expand its functionality are updated on a regular basis. Keeping up to

## INFO

Website Performance Guidelines

Document Version: 1.0

Date: 2009.09.30

date on these new versions is always recommended. To help with keeping up to date with SolSPACE Software, you can install the SolSPACE Software Update extension.

Link: [http://solSPACE.com/software/detail/solSPACE\\_software\\_update/](http://solSPACE.com/software/detail/solSPACE_software_update/)

### Design Intelligently!

Having a great team and maintaining your site's underlying software is only the foundation for building a well performing ExpressionEngine site. You also need to build your Templates intelligently so that they will do what you want while having the best performance.

Our belief is that you can accomplish great, amazing things in ExpressionEngine but the software is not going to tell you how to develop, you have to figure that out on your own. What follows are guidelines and tips to help you along this path to better performance. Remember, that while these suggestions will help, you have to balance expectations with reality. Every tag, every variable, and every conditional can add or reduce the work required to build a page. You may have to cut a feature to get the performance you want, or you may have to get extremely creative to have the content you desire.

### Solving a Problem? Simplify

Whenever you are trying to solve a problem (ex: feature not working, performance problems on a page, odd behavior), simplify the problem.

Specifically, when working in a template, copy its contents to a new template and remove everything that is not related to the problem. Focus on just a single tag or a specific aspect of the ExpressionEngine design. Narrowing it down to just 20 lines of template code is much easier than searching through an entire page looking for trouble. Try A or B testing, where you have two different templates with a simple change between them.

Once you have narrowed down a problem, use ExpressionEngine's built in debugging tools to help examine what is happening. In *Admin => Output*

## INFO

Website Performance Guidelines

Document Version: 1.0

Date: 2009.09.30

*and Debugging Preferences* turn on 'Display SQL Queries' and 'Display Template Debugging'. When logged in as a SuperAdmin and viewing the page, you will now be able to see precisely how ExpressionEngine is parsing the template and at what speed, as well as what queries are required to build the page. Often this will provide great insight into what a page's problem may be.

### Template Building: Weblog Entries

The most powerful and versatile tag in the entirety of ExpressionEngine is the *Weblog Entries* tag. Most of the major content in ExpressionEngine is stored either in a weblog or is related to a weblog's entries, so this is by far the tag that you will most often in ExpressionEngine.

A consequence of being so important is that the code for building the output for the *Weblog Entries* tag is rather complex and immense. Much is asked of it and it delivers, every single time. However, you will not always need everything it can deliver, so one of the most important parameters you will learn for reducing the load of this tag is the *disable=""* parameter ([http://expressionengine.com/docs/modules/weblog/parameters.html#par\\_disable](http://expressionengine.com/docs/modules/weblog/parameters.html#par_disable)). Learn it and use it well.

Also, if you are using the Multiple Site Manager (MSM) expansion for ExpressionEngine, always insure that you are specifying the *site=""* parameter for your *Weblog Entries* tags. If you have two weblogs in two different sites with the same short name, this will reduce the number of results ExpressionEngine has to search through to find your data. Imagine being able to halve (or more) the amount of data MySQL has to search through. A significant difference.

Further, if you do not need the full might of the *Weblog Entries* tag, consider using another tag or module to output the data you need. The Weblog module has many other tags available, and if you are only retrieving a few select pieces of data for an entry there is always the possibility that you can use the Query module to get precisely what you need with less work. Also, don't forget that you can build your own plugin that can do *precisely* what you need with the smallest amount of possible code.

## INFO

Website Performance Guidelines

Document Version: 1.0

Date: 2009.09.30

### Template Building: Embeds

ExpressionEngine gives you the ability to "embed" templates inside of other templates, allowing you to break up your design into distinct and easily reusable parts. Beware though that this feature does have a cost and may not be the most advantageous way of breaking up your design.

When a page is requested, ExpressionEngine retrieves the template for that page from the database and processes all of the template's tags, variables, and conditionals in a certain order. Then, Embeds are processed at the very **end** of this template processing; after all other variables, tags, and conditionals have been parsed and their content outputted.

For *every* single embedded template that is found, ExpressionEngine performs a query to retrieve that template and its settings. It then processes that template as if it was its own page and when finished it replaces the `{embed=""}` tag with the final, parsed output. Effectively, an embedded template is a sub-page within a page, and it requires all of the processing that building a page from a template entails. It is not a lightweight operation. So, if you are embedding a template that contains no tags or variables, you are wasting a large number of resources for very little gain.

A prime example of this is when one uses an embedded template to contain some static HTML or JavaScript with absolutely no EE tags or variables. For instances like these, we highly suggest using the *Template => Global Variables* ability in ExpressionEngine as those add virtually no load to your page load, unlike embeds.

For sanity reasons, we do not recommend multi-dimensional nesting of Embeds. This is when you have a Template with a Sub-Template with a Sub-Template. Once you start having multiple layers of Embeds in your design, debugging and development become much more complicated as it will not always be clear what Template is producing what content.

## INFO

Website Performance Guidelines

Document Version: 1.0

Date: 2009.09.30

So, use embeds wisely, and if you are embedding more than a dozen templates per page, you might wish to rethink your design.

### Template Building: Conditionals

When considering performance, there are two types of conditionals in ExpressionEngine: Simple and Advanced.

Simple conditionals can involve only three types of variables: Segment, Global Vars, and Embed. And, it is only a Simple Conditional when there is only one variable in the conditional and no *{if:elseif}* or *{if:else}* clauses.

```
{if segment_3 == ''}Show this{/if}
```

```
{if embed:entry_id != ''}Process this{/if}
```

The advantage of Simple Conditionals is that they are parsed and evaluated **prior** to any ExpressionEngine tags being processed in a template. This means that if you put ExpressionEngine tags in a simple conditional and the conditional is false, then that tag is **not** processed at all.

```
{if segment_3 != ''}
    {exp:weblog:entries entry_id="{segment_3}"}
        {title}
    {/exp:weblog:entries}
{/if}
```

Every other kind of conditional is an Advanced Conditional and is parsed near the end of Template processing and **after** tags are processed. This means that if you put a tag in an Advanced Conditional, the tag will still be processed (adding to the work required to build the page), however, the content will not be outputted to the page. So, if you can, use Simple Conditionals around tags that you may or may not want processed.

One notable exception to the Advanced Conditional rule should be mentioned. Embeds are processed **after** Advanced Conditionals, so you can

## INFO

Website Performance Guidelines

Document Version: 1.0

Date: 2009.09.30

safely put an `{embed=""}` tag in any conditional and it will be processed only when that conditional is true.

### Actions, Forms and Hashes. Oh My!

ExpressionEngine has a feature, enabled by default, called Secure Form Hashes that gives every form and Action request built by the system an extra field called XID. Do a search for "XID" in your pages and you will typically see it at least once or twice per page load.

This field contains a hash that is tied to the viewer's IP Address. When the form or Action request is submitted, ExpressionEngine checks for the submitted hash in the database and sees if it is assigned to the submitter's IP Address. The hope is that this will insure that any form or Action request will not be automatically reused by spammers or bots trying to prey on your site.

A problem can occur when you have multiple forms or Action requests in a page, as ExpressionEngine will create a unique hash for every single one. This will flood your database with new hashes on a highly trafficked site. Typically, we see this with features used for ignoring comments or reporting them as spam. If you have a hundred comments shown and also a link or form for each one, then each page load adds a 100 new hashes to the database.

Our recommendation is to find some way to build these forms or links only when they are really needed. Using AJAX or sending the user to another page with the actual form can greatly alleviate this stress on your database.

### Native Caching. The Devil's ToolBox.

ExpressionEngine has three different kinds of caching available by default: Template, Tag, and Query caching. As a general rule, we suggest you only ever use the first one, Template Caching.

The reason for this is simple. ExpressionEngine caches based on the requested URI, which means that every cache file created by ExpressionEngine is tied specifically to a specific page request (ex: `/site/comments/42/`).

## INFO

Website Performance Guidelines

Document Version: 1.0

Date: 2009.09.30

For example, if a page is requested and the template has a tag being cached, then ExpressionEngine will create a cache folder for that page's URI (ex: /site/comments/42/) and put the cache file for the cached tag in that folder. If you request the same page with just an extra, useless segment (ex: /site/comments/42/woot/) added to the URI, ExpressionEngine creates an entirely *new* cache folder for that new URI and the tag will be cached *again*. So, even if the same content is outputted for the tag on both pages, it is cached twice.

The reason ExpressionEngine does this is because it is a dynamic system with much of its processing based on what is in the URI and it cannot know intuitively what part of the URI is being used by the template or not. Good for the system, terrible for caching a tag once across the entire site. Obviously, since Template Caching caches the entire page, caching based on the URI is not a problem.

Template Caching might not always be the best idea either. Since cached files are stored on the filesystem of the server, every time a cached file is requested it must be read from the disk and loaded into PHP. The benefits of caching may be outweighed by the increase in disk activity and the time required to load the cache file. Do testing to insure Template Caching is effective for a page.

### Caching Alternatives

Solspace offers two caching alternatives to ExpressionEngine's native caching that might be helpful in reducing load.

First, the Template Morsels module. Essentially, you create a 'morsel' that is a snippet of code from your Template. The template code is then processed, cached in the database, and outputted using a simple tag. The outputted content will be the same for each page created by that template, so thus it is a truly global ExpressionEngine cache. Excellent for content that is in a sidebar or a header that is not often updated but is used everywhere.

## INFO

Website Performance Guidelines

Document Version: 1.0

Date: 2009.09.30

Further, the cached 'morsel' can be refreshed on the updating of specific Weblogs, Templates, or Categories by choosing them in the morsel's preferences. Template Morsels also includes a dynamic morsels mode. This allows you to make morsels dependent on the context of the page, allowing for more page-specific output and caching.

Link: [http://www.solSPACE.com/software/detail/template\\_morsels/](http://www.solSPACE.com/software/detail/template_morsels/)

Second, the Static Page Caching module. This module allows you to cache an *entire* page and then have any request to that page automatically redirected to the cached page, skipping ExpressionEngine altogether. If you have a specific page being hit heavily, this module allows you to create a cache of it as a Guest and display it until the traffic subsides. Since the request for the page never touches ExpressionEngine, there is no load whatsoever.

Link: [http://www.solSPACE.com/software/detail/static\\_page\\_caching/](http://www.solSPACE.com/software/detail/static_page_caching/)

## MySQL Tuning

One of the commonly neglected aspects of high traffic dynamic sites is having an amazing database server but not tuning the MySQL configuration for the server. We suggest you consult with your server administrator about tuning and optimizing it for your site. As a general guideline we suggest modifying these configuration options:

Max Connections. At least 500.

Query Cache. Enable.

Table Cache. At least 120.

Thanks.

- To Michael Boyink for reviewing and providing excellent suggestions.
- To Nevin Lyne for helping us understand file server and caching load.