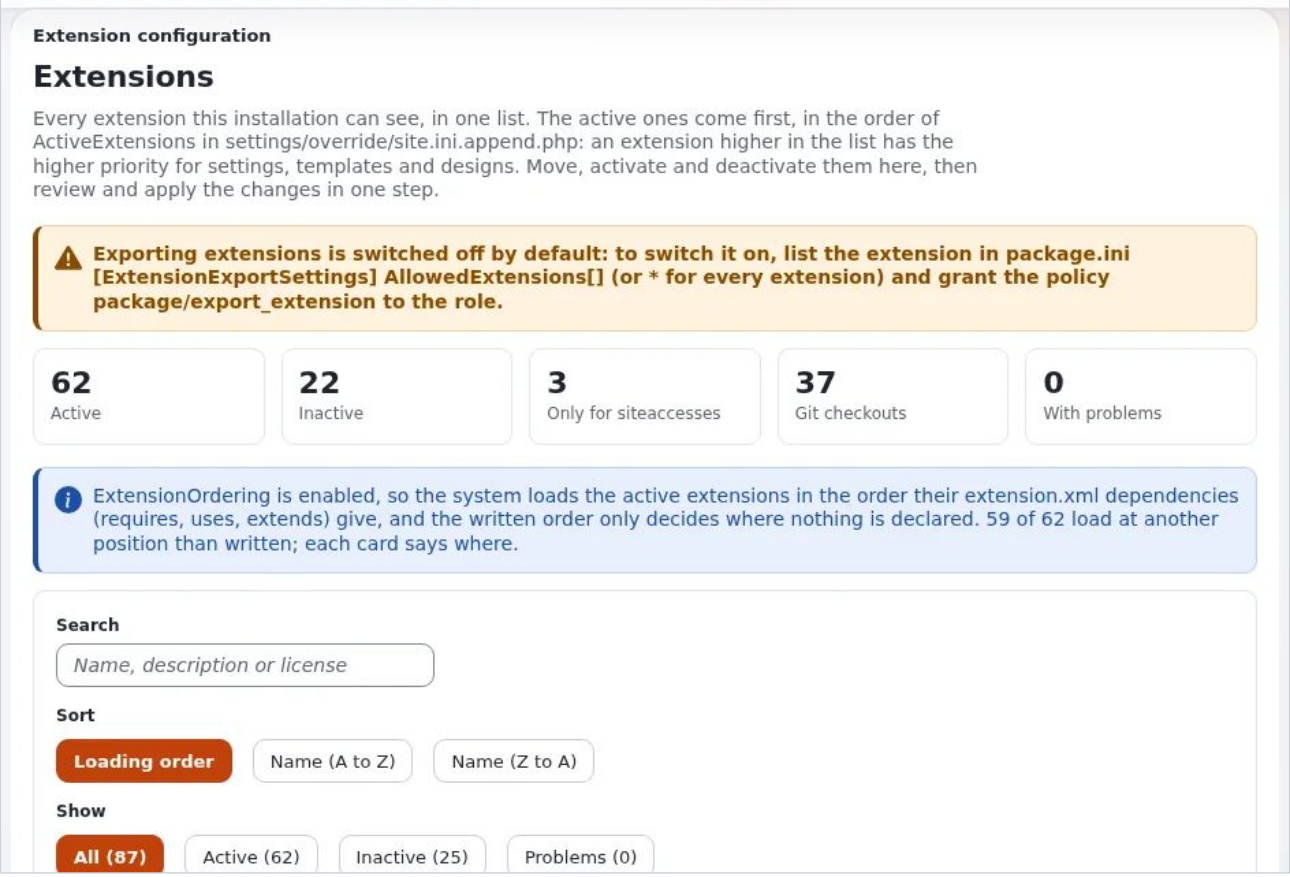
Deploying changes

How to ship a code, settings or template change to a hosted Exponential site so it is live at once and never cached half way - exp:velocity deploy and its order of steps, why a PHP-FPM reload affects other customers...

Source file: `doc/eng-us/help/hosting/operations/deploying-changes.md` Last changed 2026-11-04
https://doc.exponential.earth/exphelp/viewer/help/hosting/operations/deploying-changes.md

This page is for system administrators who ship changes to hosted Exponential sites, whether the change comes from the host (a new release, a settings change) or from a customer's developer. It answers: which steps make a change live, in which order and why, how a reload of PHP-FPM affects other customers on the same server, what is enough for a template change, and how to keep the configuration of a deployed site under version control.



Setup, Extensions: the extensions on disk, which are active, and the button to regenerate the autoload arrays

The reference is Deploying a PHP change; the walkthrough is Deploying Exponential.

Contents

1. One command for a PHP change
2. The steps and their order
3. PHP-FPM reloads on a shared server
4. Template and stylesheet changes
5. Large changes and several nodes

6. Settings per environment
7. Configuration managed by the deployment
8. Checklist

One command for a PHP change

After a changed PHP class, operator or INI file:

Shell

```
./console exp:velocity deploy --dry-run --allow-root-user   # what it would do, nothing else
./console exp:velocity deploy --allow-root-user             # the usual case
./console exp:velocity deploy --kernel --allow-root-user    # a kernel class was added or renamed
```

A real run prints PASS, FAIL or SKIP and the time for each step, and "stops at the first failure with exit status 1". Despite its name the command serves every setup: `--no-velocity` for a site served by Apache or nginx alone, `--no-fpm` for one served by Velocity alone.

Option	Use it when
`--kernel`	a kernel class was added or renamed (kernel autoloads, restricted to `kernel/` and `lib/`)
`--fpm-reload`	`php.ini` or the pool changed, a PHP extension was added, PHP files were removed or renamed in place
`--no-fpm`, `--no-velocity`, `--no-autoload`	skip that part
`--packer`	an `ezjscServer_*` function or its settings changed: also clear packed scripts and styles
`--rebuild-phar`	force the rebuild of Velocity's engine archive

The steps and their order

#	Step
1	extension autoloads
2	kernel autoloads, only with `--kernel`
3	INI caches
4	template, template-override, translation and design_base caches
5	Velocity's engine archive, when a file in `kernel/`, `lib/` or `autoload/` changed (every file parsed first)
6	PHP-FPM: wait until OPcache has revalidated the changed files, or reload it when that cannot be relied on
7	restart Velocity, when it is running
8	content, HTTP cache and template-block caches (and the packer with `--packer`)
9	Velocity's response cache
10	warm-up: each `[DeploySettings] WarmUpURL[]` requested once by a background process

"The order is the point." The caches that hold rendered output are cleared only after PHP-FPM and Velocity run the new code; "cleared before, every page requested in between is rendered

by the old code and cached again, and the change looks as if it had not worked." A file that does not parse fails step 5, before any running server is touched.

Set warm-up addresses per installation, so the first visitor does not rebuild the caches:

INI

```ini
# settings/override/velocity.ini.append.php
[DeploySettings]
WarmUpURL[]=https://www.example.com/
WarmUpURL[]=https://www.example.com/admin/user/login
```

PHP-FPM reloads on a shared server

`[DeploySettings] PhpFpmService=auto` finds the PHP-FPM service that serves the installation (it looks for the pool named after the installation's hosting domain), so the right one is reloaded on a server with several PHP versions. "On a Plesk server, for example, `systemctl restart php-fpm` restarts a PHP no site uses" (Deploying, 3.4).

> **Warning**
>
> A reload is not confined to one customer. "The master asks every child of every pool to finish its request and starts no new child until the last one has." Every request that arrives meanwhile, on any site of that PHP version, waits for the longest request still running. That is why `[DeploySettings] PhpFpmReload=auto` (shipped) does not reload when the pool's OPcache revalidates files by itself (`opcache.validate_timestamps` on, `revalidate_freq` at most 60 seconds): it waits that long instead.

For a host this means: keep `PhpFpmReload=auto`, configure OPcache revalidation in the pools (OPcache), and schedule changes that need `--fpm-reload` (a `php.ini` or pool change) as server maintenance, not as one customer's deploy. Giving each customer its own pool limits who waits, but not the master's reload (Per-customer users and pools).

The reload needs root; run as anyone else, the step is skipped with a note naming the service.

Template and stylesheet changes

Edits to a template or a stylesheet need less: clear the caches without a restart.

Shell

```shell
./console exp:cache clear --id=template --allow-root-user
./console exp:velocity cache clear --apply --allow-root-user
```

After an extension is activated, its templates are not found until the design_base cache is cleared: the page "answers 200 with an empty content area". `exp:velocity deploy` clears it, with the template-override cache, in step 4 whenever settings or templates changed since the last deploy (activating an extension changes settings); `--all-caches` clears it in any case. A deploy of PHP classes only keeps the INI caches and the pages already rendered: `--clear-content` clears the rendered pages anyway. For stylesheets packed by ezjscore, deploy with `--packer`. "Never run only `exp:cache clear --all` and then forget the web server: FPM and Velocity keep the old classes in memory until they are reloaded."

Large changes and several nodes

- **Maintenance.** `exp:velocity deploy` does not switch maintenance mode on or off. "For a deploy that changes the database or many files at once, open a window first and close it after the smoke test" (Maintenance mode).
- **Several nodes.** Run the deploy on each node (Several web nodes).
- **No shell.** For customers without shell access, the extension `git_manager` offers a git dashboard and backups in the administration (git_manager); the cache and reload steps still have to happen afterwards.

Settings per environment

One installation often exists as a live site, a staging copy and a developer copy. `EXP_ENV` names the environment of a machine in `config.env.php` (copied from `config.env.php.example`, kept out of version control), and every settings file can have a variant for it next to it, such as `settings/override/site.ini.test.append.php` (Settings per environment). Mail transport, site addresses and debug settings then differ per machine without hand edits after each deploy.

Configuration managed by the deployment

On a site whose configuration lives in version control, edits made in the administration land in the release directory, "where version control does not see them and the next deployment drops them". One switch per siteaccess prevents that (Configuration managed by the deployment):

INI

```
# settings/siteaccess/<admin siteaccess>/site.ini.append.php
[ConfigurationSettings]
ManagedByDeployment=enabled
```

With it, the settings editor, the toolbar and menu editors, the template editor, the setup wizard, extension activation, the code generators and package upload, install and removal answer like a disabled view, and `eZINI::save()` refuses in a web request. Command-line scripts and the deployment itself still write. It ships `disabled`. "File permissions on the release directory remain the stronger measure."

> **Tip**
>
> Offer `ManagedByDeployment=enabled` as the default of a managed plan in which the host or the customer's developers deploy from a repository; leave it off on plans where customers configure their site in the browser.

Checklist

- ☐ Every PHP change ends with `exp:velocity deploy` (with `--no-velocity` or `--no-fpm` as the setup needs).
- ☐ `PhpFpmService` finds the right service; `PhpFpmReload=auto` is kept and OPcache revalidates in the pools.
- ☐ `WarmUpURL[]` names the front page and the admin login of each site.

☐ Template changes clear the template caches and Velocity's response cache.

☐ Large deploys run in a maintenance window; multi-node deploys cover every node.

☐ Environment variants keep staging from acting like live; deployed sites use `ManagedByDeployment` where it fits.

Related pages

- Upgrading customer sites
- Cache clearing and warming
- PHP-FPM tuning
- Velocity for hosts
- Composer and git on servers
- Migrations on hosted sites, Migrations
- Free documentation: this page is free to copy, translate and improve under the GNU FDL 1.3.

> **Become an Exponential Partner**
>
> Partners share their information, provide funding, contribute back and develop the products with us, and that is how Exponential grows. A partner website and dashboard are planned for mid-2027: please become a partner today to speed up our growth rate and land new features faster.
>
> Partners: start here | How to apply

Exponential Copyright © 1998-2026 Exponential Foundation and 7x
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".
This documentation is free documentation, licensed under the GNU Free Documentation License 1.3 or later:
https://www.gnu.org/licenses/fdl-1.3.html.