Deploying Exponential: Apache with PHP-FPM, Velocity or FrankenPHP

Source file: doc/eng-us/guides/deploying.md Last changed 2026-11-04
https://doc.exponential.earth/exphelp/viewer/guides/deploying.md

This guide is for the person who puts a site on a server: an operator, or a developer who runs their own. It takes a freshly installed Exponential site to a server that visitors reach over HTTPS. Then it shows how to ship a code change safely and how to keep the site fast. Read it top to bottom. Every command exists in the installation; where a result is shown, it is what the command prints.

> **Warning**
>
> `.htaccess` **slows down every request.** When `.htaccess` files are enabled (`AllowOverride` anything but `None`), let alone used, Apache looks for them and reads and evaluates them on every request, in every directory of the path, and page loading times go up drastically (measured). The only real fix is to switch the feature off for the virtual host: `AllowOverride None` in the HTTPS virtual host, and the rules in the virtual host's own configuration as mod_rewrite rules. That is far faster in production speed tests and well worth the time it takes to optimize a production configuration. Switch over right away: copy the rules of the example HTTPS virtual host into your virtual host (with the static cache, also `staticcache_offload.conf`), set `AllowOverride None`, run `apachectl configtest`, reload Apache and test the site. The same rules for other servers: nginx, LiteSpeed, Caddy, IIS, Velocity (built in); where virtual host rules go in a control panel: Plesk, cPanel, DirectAdmin, ISPConfig. **On LiteSpeed or OpenLiteSpeed,** the server's own page cache can also answer anonymous visitors and crawlers without PHP: Page caching with LiteSpeed and OpenLiteSpeed.

You need a Linux server, shell access in the installation directory (the folder that holds `index.php` and `console`), and a site that already runs. If you have no site yet, run `./console exp:install --apply` first (Getting started, Installing in one command).

Commands use `./console`, the shortcut for `bin/php/console`. The examples add `--allow-root-user` because they are run as root; leave it off when you run as the site user.

1. Choose how the site is served

Exponential runs behind one of three setups. Pick one for now; you can switch later without touching content.

Setup	Choose it when	Time to a running site
Apache (or nginx) with PHP-FPM	the server already has a web server and a hosting panel, or you want the classic setup	10 minutes
Exponential Velocity (the qbix engine)	you want the fastest engine, persistent workers, a built-in response cache and no web server to configure	2 minutes
FrankenPHP	you want one verified binary with TLS and HTTP/2 and no PHP on the machine to maintain	5 minutes

Velocity is the recommended engine for every stage. The three can run side by side on different ports, so trying one never takes the others down (engines compared).

2. Check the machine before you start

Shell

```
php -v | head -1
./console exp:velocity ext check --allow-root-user
```

The first line shows the PHP version (8.0 or newer, for Velocity too). The second lists every PHP extension the installation wants, marks what is missing and prints the install command for your system, for example `sudo dnf install -y php-odbc`. Install what it reports for the features you use, then restart PHP-FPM or the server so the extensions load. Missing optional extensions do not stop the site.

3. Option A: Apache with PHP-FPM

3.1 Point the web server at the installation

The document root is the installation directory itself. Exponential keeps its own rewrite rules in a file you enable by copying:

Shell

```
./console exp:setup:htaccess --apply      # or by hand: cp .htaccess_root .htaccess
```

The command shows what it would do without `--apply`, copies with the template's mode (and, run as root with `--allow-root-user`, its owner), and refuses with a diff when a `.htaccess` that differs is already there; `--force --apply` then replaces it after a backup to `var/backup/setup/<time>/`. `./console exp:setup:config --apply` puts `config.php` into place the same way, and `./console exp:config:set <NAME> <value> --apply` changes one of its settings. See Setting up .htaccess and config.php from the console, which also has a provisioning script for containers and CI.

The copy turns on `.htaccess` based virtual host mode: dot files, `.git` and scripts below the front controllers answer 404, `/api/` goes to `index_rest.php`, static design files are served directly and everything else goes to `index.php`. Do not edit security rules out of it to make something work (security defaults).

A minimal Apache virtual host for PHP-FPM (adjust the paths and the socket to your system):

Config

```
<VirtualHost *:80>
    ServerName example.com
    DocumentRoot /var/www/example.com
    DirectoryIndex index.php

    <Directory /var/www/example.com>
        Options FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    <FilesMatch \.php$>
        SetHandler "proxy:unix:/run/php-fpm/www.sock|fcgi://localhost"
    </FilesMatch>
</VirtualHost>
```

`AllowOverride All` is what lets Apache read the `.htaccess` you just created, and `mod_rewrite`, `mod_proxy` and `mod_proxy_fcgi` must be loaded. Check and reload:

Shell

```
apachectl configtest      # Syntax OK
systemctl reload apache2  # or httpd on Red Hat style systems
```

3.2 Test it

Shell

```
curl -sI http://example.com/ | head -1
```

Expect `HTTP/1.1 200 OK` (or a redirect to the first page). If you get 500, open the PHP-FPM and Apache error logs; if you get 404 for every page, `.htaccess` is not being read (`AllowOverride`) or `mod_rewrite` is off.

3.3 HTTPS

Obtain a certificate with your usual tool (certbot, your hosting panel) and add a `*:443` virtual host with `SSLEngine on`, `SSLCertificateFile` and `SSLCertificateKeyFile`, the same `Directory` block and the same `SetHandler`. Test with `curl -sI https://example.com/ | head -1`.

3.4 After every PHP change: reload the right PHP-FPM

On a server with several PHP versions, the FPM master that serves your site is not always the one named `php-fpm`. On a Plesk server, for example, `systemctl restart php-fpm` restarts a PHP no site uses. `exp:velocity deploy` (section 7) finds the right one for you; the setting is `[DeploySettings] PhpFpmService`:

Shell

```
./console exp:velocity config get DeploySettings PhpFpmService --allow-root-user
```

It prints `auto`, which looks for the pool named after the installation's hosting domain and reloads its service.

4. Option B: Exponential Velocity (the `qbix` engine)

Velocity is a PHP application server that ships with Exponential. It needs no Apache, nginx or FPM.

4.1 Select the engine and start it

Shell

```
./console exp:velocity config set --apply ServerSettings Engine qbix --allow-root-user
./console exp:velocity start --allow-root-user
./console exp:velocity status --allow-root-user
```

`config set` writes `settings/override/velocity.ini.append.php`, the one file for this installation's settings (the shipped defaults stay in `settings/velocity.ini`). `status` lists the addresses it serves, the number of processes, whether HTTPS is on and where the logs are:

```
Process    52 processes, parent ... · port 8088, 8080
HTTPS      on, port 8080
Access     var/vc/qbix/log/site-access.log
Errors     var/vc/qbix/log/site-error.log
```

Plain HTTP is on `Port` (8088 shipped), HTTPS on `HTTPSPort` (8080 shipped), both bound to `Host` (`127.0.0.1` shipped). Open `http://127.0.0.1:8088/` on the same machine. To serve the network, set `Host`:

Shell

```
./console exp:velocity config set --apply ServerSettings Host 0.0.0.0 --allow-root-user
./console exp:velocity restart --allow-root-user
```

Ports below 1024 need root; the usual setup is Velocity on its own ports behind a hosting panel or a firewall rule, or a front web server proxying to it.

4.2 Stop, restart, reload without dropping requests

Shell

```
./console exp:velocity graceful --allow-root-user    # re-exec, listening socket kept
./console exp:velocity restart --allow-root-user
./console exp:velocity stop --allow-root-user
```

4.3 Look around

Shell

```
./console exp:velocity layout --allow-root-user      # every file the server uses
./console exp:velocity config list ServerSettings --allow-root-user
```

`layout` shows the configuration tree in the style of Apache on Debian (`/etc/vc` with `sites-available`, `sites-enabled`, `conf-available` and so on). Sites, snippets and modules are switched on and off like `a2ensite`: `./console exp:velocity site enable --apply <name> --allow-root-user` (on-disk layout). The server also has a control panel (Velocity control panel).

Depth: Velocity web server, persistent workers, worker pool, engine settings.

5. Option C: FrankenPHP

FrankenPHP is the Caddy web server with PHP built in, as one file. Exponential downloads a pinned release and checks its SHA-256 before running it.

Shell

```
./console exp:velocity install --engine=frankenphp --allow-root-user
./console exp:velocity start --engine=frankenphp --allow-root-user
./console exp:velocity status --engine=frankenphp --allow-root-user
```

By default it serves plain HTTP on port 8089 and HTTPS on 8444, so `https://` works from the first start (a self-signed certificate stands in until you give it a real one). Useful variants:

Shell

```
./console exp:velocity start --engine=frankenphp --no-https --allow-root-user   # plain HTTP, thi
s start only
./console exp:velocity install --engine=frankenphp --check --allow-root-user    # re-hash the ins
talled binary
```

PHP runs in classic mode (clean state per request, as under PHP-FPM). The engine has no response cache yet; if you rely on one, use Velocity. Two things bite on a first install: the embedded PHP has no machine `php.ini`, so set the database socket under `[PHPSettings] IniOptions[]` if your database is reached over a socket, and a re-published release can make the pinned checksum stale. The complete tested walk-through, with both fixes, is FrankenPHP as the Exponential web engine.

To run all three engines side by side, give each its own port and use `--engine=<name>` or `--all` on the control commands.

6. HTTPS

Setup	How
Apache	the certificate in the *:443 virtual host (section 3.3)
Velocity	`[HTTPSSettings]` in `velocity.ini` and the certificate subsystem
FrankenPHP	on by default; `[HTTPSSettings] Certificate` and Key supply your own

For Velocity with a certificate you already have (a PEM file with certificate and key; both settings may name the same file):

Shell

```
./console exp:velocity config set --apply HTTPSSettings Certificate /etc/ssl/example.com/fullchai
n.pem --allow-root-user
./console exp:velocity config set --apply HTTPSSettings Key /etc/ssl/example.com/privkey.pem --al
low-root-user
./console exp:velocity config set --apply HTTPSSettings Enabled true --allow-root-user
./console exp:velocity restart --allow-root-user
curl -skI https://127.0.0.1:8080/ | head -1
```

Expect `HTTP/2 200` (or a redirect). The Velocity server can also make a self-signed certificate on its own, obtain and renew Let's Encrypt certificates and swap a renewed certificate in without a restart: Velocity HTTPS and certificates.

Strict-Transport-Security is sent on HTTPS answers with `[HTTPSSettings] HSTSMaxAge`

(shipped as 0, which sends none; start with 300). Raise it to a year (`31536000`) only when every name of the site is served over HTTPS for good: a browser remembers the value even after you lower it (HTTP/2 and security).

7. Ship a PHP change with one command

After you change a PHP class, an operator or an INI file, run:

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
```

The dry run prints the fourteen steps, each marked `DRY`:

```
[ 1/14] DRY   extension autoloads: php bin/php/ezpgenerateautoloads.php -e -q
[ 2/14] DRY   INI caches: php bin/php/ezcache.php --clear-tag=ini --allow-root-user
...
[ 8/14] DRY   PHP-FPM sees the new code: wait 11 s (no reload)  -- <pool>: OPcache revalidates ev
ery 10 s
[ 9/14] DRY   restart Velocity: php bin/php/velocity.php restart
[10/14] DRY   content cache ...
[13/14] DRY   Velocity response cache: exp:velocity cache clear
[14/14] DRY   warm-up requests: curl 2 URLs in the background
```

A real run prints PASS, FAIL or SKIP and the time for each step, and stops at the first failure with exit status 1. The order matters: the caches that hold finished pages are cleared last, after PHP-FPM and Velocity run the new code, so a page rendered by old code is never cached again. PHP-FPM is reloaded only when its OPcache does not revalidate files by itself: a reload stalls every pool of that PHP version until its longest request ends, so the step waits for OPcache instead (`[DeploySettings] PhpFpmReload`). The last step requests `[DeploySettings] WarmUpURL[]` in the background, so the first visitor finds the caches built. Options: `--no-fpm` (Velocity only), `--fpm-reload` (reload anyway: php.ini or pool changed, files renamed), `--no-velocity` (Apache only), `--no-autoload`, `--rebuild-phar`, `--packer` (also clear packed scripts and styles, when an `ezjscServer_*` function changed). Details and every step: Deploying a PHP change.

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

Shell

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

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.

8. Make it fast

Work down this list; each item is a switch or a command that exists, and `exp:cache status` shows where you stand.

Shell

```
./console exp:cache status --allow-root-user
```

It prints one line per layer, for example `httpcache the HTTP cache is on: ... entries`, `querycache the SQL query cache is on`, `velocity response cache ...: N files`, and `php OPcache not enabled for this server` when the opcode cache is off for the command line (normal; it matters for the web process).

1. **Opcode cache.** On PHP-FPM keep `opcache.enable=1` in the pool's PHP configuration. Velocity passes its own opcode cache options to the server process; the shipped values are in `[PHPSettings]` of `settings/velocity.ini`. Measured effect and how to profile: Velocity and the opcode cache.
2. **Response cache (Velocity).** `[CacheSettings] Enabled=enabled`, `DefaultTtl=30` seconds. A page served from it never reaches PHP. Clear it after a template change with `./console exp:velocity cache clear --apply --allow-root-user`. See Velocity response cache.
3. **HTTP cache.** The role-aware page cache for any engine: `./console exp:cache httpcache status --allow-root-user` (HTTP cache).
4. **SQL query cache.** On by default; `./console exp:cache querycache status --allow-root-user` (SQL query cache).
5. **Static files.** Velocity serves them with a long `StaticMaxAge` and can precompress them (`[ServerSettings] StaticMaxAge`, `PrecompressStatic`) (static files and images).
6. **Static cache.** Whole pages written as files for Apache to serve; shipped off in the installer. `exp:cache static status` says whether it is on (static cache generator, defaults).
7. **Workers.** `[ServerSettings] Workers` (4 shipped) is the Velocity process count; raise it with the number of CPU cores and measure again after each change.

Measure with the same request before and after; a single page load proves nothing:

Shell

```
curl -s -o /dev/null -w '%{time_total}s\n' http://127.0.0.1:8088/
```

Run it three times: the first request fills the caches, the others show the cached time.

9. Check that you are done

- `./console exp:velocity status --allow-root-user` (Velocity) or `curl -sI` on your address (Apache) answers 200.
- The browser shows the padlock on the HTTPS address.
- `./console exp:velocity deploy --dry-run --allow-root-user` lists a `reload PHP-FPM` step with the right service.
- `./console exp:cache status --allow-root-user` shows the layers you expect.
- Apache: `./console exp:setup:status --allow-root-user` says `.htaccess` is a copy of `.htaccess_root` (or of `.htaccess_root_static`), and `./console exp:config:list --only-set --allow-root-user` shows the `config.php` settings you expect.

Settings used in this guide

All are in `settings/velocity.ini` (shipped defaults). `exp:velocity config set` writes your values to `settings/override/velocity.ini.append.php`; the scope is the installation.

Block	Key	Default	What it does
`[ServerSettings]`	`Engine`	`php`	The engine `exp:velocity` starts when no `--engine` is given: `php`, `qbix` or `frankenphp`
`[ServerSettings]`	`Host`	`127.0.0.1`	Address Velocity (`qbix`) listens on; `0.0.0.0` for the network
`[ServerSettings]`	`Port`	`8088`	Plain HTTP port of Velocity
`[ServerSettings]`	`HTTPSPort`	`8080`	HTTPS port of Velocity
`[ServerSettings]`	`Workers`	`4`	Number of Velocity worker processes
`[ServerSettings]`	`StaticMaxAge`	`31536000`	Cache lifetime in seconds sent with static files
`[ServerSettings]`	`PrecompressStatic`	`enabled` (code default, not listed in the file)	Keep compressed copies of static files
`[CacheSettings]`	`Enabled`	`enabled`	Velocity's response cache
`[CacheSettings]`	`DefaultTtl`	`30`	Seconds a response stays in that cache
`[HTTPSSettings]`	`Enabled`	`false`	HTTPS on Velocity's `HTTPSPort`
`[HTTPSSettings]`	`Certificate`, `Key`	empty	PEM files of your certificate and key
`[HTTPSSettings]`	`HSTSMaxAge`	`0`	Strict-Transport-Security lifetime in seconds
`[DeploySettings]`	`PhpFpmService`	`auto`	The PHP-FPM service `exp:velocity deploy` reloads
`[FrankenPHPSettings]`	`Port`, `HTTPSPort`	`8089`, `8444`	Ports of the FrankenPHP engine
`[PHPServerSettings]`	`Port`	`8087`	Port of the php engine (PHP's own server)

Related pages

- Features: Velocity web server, Velocity HTTPS and certificates, Installing in one command, Console commands for .htaccess and config.php.
- Guide: Setting up .htaccess and config.php from the console, with a provisioning script for containers and CI.
- Specifications: Velocity engine settings, worker pool, HTTP/2 and security.
- Measuring the result: Benchmarking Exponential (Apache against Velocity, a deploy checked for regressions).
- Upgrade notes: Velocity engines, FrankenPHP, Velocity engine upgrade notes, cache console .
- History: Velocity chronicle, October 2026.
- Deploying with migrations, Migrations

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.