Installing Exponential 6.0

Source file: doc/eng-us/INSTALL.md Last changed 2026-11-07
https://doc.exponential.earth/exphelp/viewer/INSTALL.md

This guide installs Exponential 6.0 from nothing to a working site: what you need, how to get the code, the three ways to install (the browser setup wizard, the kickstarter, the one-command console install), how to serve the site and what to do afterwards. For what Exponential is and how it is served, read the project README first. For a guided first hour after the install, read Getting started.

> **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.

Every command is run from the installation root: the directory that holds `index.php`, `autoload.php`, `bin/` and `settings/`. The command console is `bin/php/console`. Many installations add a shortcut to it in the root (`ln -s bin/php/console console`), so `./console` and `php bin/php/console` are the same program. When you run a script as `root`, add `--allow-root-user`; the script says so when it is needed.

> **The full book.** This page is the short guide. Each section below has a chapter in the installation book with the complete detail, annotated examples and references: the setup wizard page by page, every kickstart.ini key, Exponential Velocity serving HTTPS without a separate web server, SQLite 3 under heavy production load, the operations handbook, troubleshooting and security hardening.

Contents

1. What you get, and the quick start
2. Requirements
3. Get the code
4. Choose an install method
5. Install with the setup wizard

6. Install with the kickstarter
7. Install with one console command
8. Serve the site: web server, HTTPS and permissions
9. SQLite in production
10. After installing
11. Upgrading from an older version
12. Troubleshooting

1. What you get, and the quick start

Full chapter: 01-introduction

An installation is the Exponential kernel, the extensions listed in `composer.json` (which ones it has and activates: Extensions of a new installation), a site package with its content and design, and three siteaccesses. Their names are always `site`, `admin` and `editor`; what you choose during the install is the URL path, port or host name that leads to each (the match value):

Siteaccess	For	Default match value with `exp:install`
`site`	visitors	the path `/site` (or port 8080, or the host you give)
`admin`	administrators	the path `/admin` (or port 8081, or the admin host you give)
`editor`	editors: the administration for content only, without setup, design and developer tools	the path `/editor` (or the public port + 2, or `edit.<public host without www.>`)

The shortest path that works needs only PHP and Composer. It installs on SQLite (no database server) and serves the site with the PHP server that Exponential can start for you:

Shell

```
composer create-project se7enxweb/exponential exponential
cd exponential
php bin/php/console exp:install --apply --random-password --url=http://localhost:8087
php bin/php/console exp:velocity start --engine=php
```

Open `http://localhost:8087/site/` for the site and `http://localhost:8087/admin/user/login` for the administration. The user is `admin`; the password is printed once at the end of the install and written to `var/log/initial-admin-password` (readable by the owner only). Change it after the first login and delete that file. Stop the server with `php bin/php/console exp:velocity stop --engine=php`.

The `php` engine is PHP's own server, for development. For a real site use Exponential Velocity's `qbix` engine or Apache/nginx with PHP-FPM (section 8).

Building a website rather than trying one? Get the code as Git clones with their full history instead: the same two install commands work in a clone, and section 3 shows the commands that make the kernel and every extension a clone you can follow, change and contribute back from.

2. Requirements

Full chapter: 02-requirements

PHP

Version	**PHP 8.0 to 8.5**. `composer.json` declares `^7.4 || ^8.0` (and accepts later 8.x). PHP 8.0 is the oldest supported version, so the stock PHP of Red Hat Enterprise Linux 9 (and AlmaLinux 9, Rocky Linux 9) runs Exponential under Apache, PHP-FPM or Velocity. Use the newest PHP your system offers. See PHP 8.0 support.
Exponential Velocity	part of Exponential as of 6.0.15: `composer.json` requires `se7enxweb/exponential-velocity ~0.0.4.57` (through `se7enxweb/exponential-php-floor`, on PHP 8.0 and later), so `composer install` installs it (Velocity is part of Exponential 6.0.15). It runs on **PHP 8.0 to 8.5** from its release 0.0.4.54 (until then it needed 8.1), with the `sockets` extension (`pcntl` and `openssl` for the worker pool and HTTPS). On RHEL 9 the stock PHP 8.0 runs it; see PHP 8.0 support.
Release tags	`v6.0.8` to `v6.0.14` require PHP 8.1; on PHP 8.0 install the 6.0.15 line, which accepts 8.0: `~6.0.15` (the tag `v6.0.15` once it is published), or the `main` branch that carries it (`dev-main`, which Composer also knows as `6.0.x-dev`).
64-bit PHP	needed for more than 30 languages (`composer.json`, `suggest`).
`memory_limit`	at least `64M` (setup check `memory_limit`). Restart the web server or PHP-FPM after editing `php.ini`.
`max_execution_time`	at least 30 seconds (setup check `execution_time`).
`date.timezone`	must be set in `php.ini` (or in `.htaccess`); see the list of timezones.
File uploads	`file_uploads` on (setup check `file_upload`).

PHP extensions:

Needed by	Extensions
`composer.json` (`require`)	`dom`, `libxml`, `mbstring`, `pcre`, `json`, `iconv`, `reflection`, `session`, `spl`, `simplexml`
the setup wizard's critical checks (`settings/site.ini [SetupSettings] CriticalTests`)	also `zlib`, `intl`, `xsl`; one database extension (`sqlite3`, `mysqli`, `pgsql`, `mongodb` or `oci8`); `gd` or the ImageMagick `convert` program for images
recommended (`composer.json`, `suggest`)	`curl` (talking to other servers, packages over SSL), `gd`, `openssl` (secure random bytes), `pcntl` (asynchronous publishing)

The setup wizard's system check tests all of these and tells you what is missing. On a running installation, `php bin/php/console exp:velocity ext check` lists every extension the installation wants, marks what is missing and prints the install command for your system.

Composer

Composer 2.x. It installs the kernel's libraries, the expcomponents (`se7enxweb/expcomponents-*`, Exponential's fork of the Zeta Components, since 6.0.15), and the Exponential extensions that `composer.json` lists.

Database

> **Supported databases (Exponential 6.0.x), in order of preference:** 1. SQLite 3 (the default) · 2. PostgreSQL · 3. MongoDB · 4. MariaDB · 5. Oracle · 6. MySQL. Microsoft SQL Server is known to be possible with additional work at this time. Details: Database support.

Database	Driver (`DatabaseImplementation`)	PHP extension	Notes
1. SQLite 3	`sqlite3`	`sqlite3`	The default. No database server: the database is one file under `var/storage/sqlite3/`. A full production database, see section 9.
2. PostgreSQL	`pgsql` (also `ezpostgresql`, `postgresql`)	`pgsql`	Needs `pgcrypto` in the database.
3. MongoDB	`mongodb`	`mongodb` and the library `mongodb/mongodb`	PHP 8.2 or later.
4. MariaDB	`mysqli` (also `ezmysqli`, `mysql`)	`mysqli`	UTF-8 (`utf8mb4`) is required.
5. Oracle 19c or later	through the `ezoracle` extension	`oci8`	`ezoracle` must be in `extension/`; it is a dependency in `composer.json`.
6. MySQL	`mysqli` (also `ezmysqli`, `mysql`)	`mysqli`	UTF-8 (`utf8mb4`) is required.

The names are the aliases in `settings/site.ini`, block `[DatabaseSettings]`, `ImplementationAlias[]`. The minimum server versions the wizard checks are in `kernel/setup/ezsetupcommon.php` (MySQL 4.1.1, PostgreSQL 8.0, SQLite 3.0.1, MongoDB 4.0, Oracle 19); the practical minimums the current drivers need are higher (SQLite 3.33, PostgreSQL 13, MongoDB 6.0), see Database support. Exponential runs on almost any operating system with PHP 8.0 or later (PHP 7.4 usable without support): see Operating system support.

Web server

One of these, chosen in section 8:

- **Exponential Velocity**, the recommended application server: it serves HTTP and HTTPS itself, with its own certificates, and needs no other web server.
- **Apache** with `mod_rewrite` and PHP-FPM (or mod_php). The rewrite rules ship in `.htaccess_root`.
- Any other server that can rewrite a URL to `index.php` (nginx, LiteSpeed, Caddy and others, listed in the README). This guide gives no configuration for them.

3. Get the code

Full chapter: 03-getting-the-code

Pick one. All give the same installation. This installs Exponential 6; the Symfony based distributions (Exponential Platform Legacy, Nexus and DXP) and the site packages to choose from are compared in Distributions of Exponential.

Git clones with history (recommended for anyone who builds websites): the kernel and every se7enxweb extension arrive as Git clones, so you can see what changed, follow development, keep your own changes on a branch and send fixes back as pull requests. Develop with Git explains it step by step.

Shell

```
composer config --global 'preferred-install.se7enxweb/*' source
git clone https://github.com/se7enxweb/exponential.git
cd exponential
git switch -c my-site
composer install
```

The first line is needed once per user account: it makes Composer clone every `se7enxweb/*` package instead of downloading its release archive.

Composer project (quickest for a site you only run; release archives without history):

Shell

```
cd www_root_directory
composer create-project se7enxweb/exponential exponential
cd exponential
```

Add a version to pin a release, for example `composer create-project se7enxweb/exponential:v6.0.14 exponential`. Add `--prefer-source --keep-vcs` to get Git clones from `create-project` too.

Composer require into an empty directory (the package from Packagist):

Shell

```
mkdir exponential
cd exponential
composer require se7enxweb/exponential
```

Then check that the console runs:

Shell

```
php bin/php/console --version
```

The first line names the version, for example `console (Exponential) 6.0.15stable`.

Exponential Velocity. `composer.json` requires it at `~0.0.4.57`, so `composer install` installs it on every supported PHP, 8.0 included. Until its release 0.0.4.54 it needed PHP 8.1 and was only suggested; an installation made in that time adds it with:

Shell

```
composer require se7enxweb/exponential-velocity:~0.0.4.57
```

Why this matters, and what it brings: Velocity is part of Exponential 6.0.15.

4. Choose an install method

Full chapter: 04-choosing-an-install-method

All three methods run the same setup steps (`kernel/setup/steps/`) and produce the same kind of installation.

	Setup wizard	Kickstarter	Console install
Started by	opening the site in a browser	`php bin/php/console exp:kickstarter run --apply --force`	`php bin/php/console exp:install --apply`
Answers come from	the forms, one page per step	`kickstart.ini` in the installation root	command-line options, every one with a default
Best for	a first install, trying Exponential	repeatable, scripted or unattended installs	the fastest install; scripts and CI
Needs a web server during install	yes	no	no
Dry run	no	`run --dry-run`	`--dry-run`, `--print`
Protects an existing installation	no (it only starts while `CheckValidity=true`)	needs `--force` for the step that writes the database	refuses unless `--force`

How the site is served is chosen separately (section 8):

	Exponential Velocity (`qbix`)	Apache or nginx with PHP-FPM	FrankenPHP (through Velocity)
Other web server needed	no	yes	no
HTTPS	built in: an existing certificate, Let's Encrypt or any ACME CA, self-signed fallback	the web server's, certificate from your own tool	built in, self-signed when no certificate is set
PHP	8.0+	8.0+	the binary's own
Role	**recommended**, every stage from development to production	the classic setup, shared hosting	production-ready alternative

5. Install with the setup wizard

Full chapter: 05-setup-wizard

How it starts

`settings/site.ini` ships with `[SiteAccessSettings] CheckValidity=true`. While it is `true`, every request is sent to the setup wizard (module `setup`, function `init`) instead of the site. The last step writes `CheckValidity=false` into `settings/override/site.ini.append.php`, and from then on the site is served.

1. Serve the installation with a web server (section 8), for example `php bin/php/console exp:velocity start --engine=php`.
2. Open the site address in a browser. The wizard's first page appears.

While the wizard runs, only the browser that started it gets through (it carries the cookie `exp_setup_wizard`); every other visitor sees a "The site is being set up" page (HTTP 503). The hold ends with the wizard's last page, or 30 minutes after the wizard's last request (`expMaintenance::WIZARD_LEASE`).

If a `kickstart.ini` is in the installation root, the wizard reads it: steps whose section has `Continue=true` are skipped, the others are pre-filled (section 6). Remove or move `kickstart.ini` if you want to answer every page yourself.

The steps

The order is the step table in `kernel/setup/steps/ezstep_data.php`.

#	Page	What to enter
1	Welcome	Nothing; continue.
2	System check	The critical checks: directory permissions, PHP version, a database extension, image conversion, memory limit, execution time, `allow_url_fopen`, sessions, file uploads, `zlib`, `dom`, `iconv`, `mbstring`, `intl`, `xsl`, the timezone and the kernel's libraries (the expcomponents, earlier the Zeta Components). Each problem comes with the fix, for permissions the exact chmod/chown commands. Fix them and press **Next** to run the check again. A test can be ignored with its checkbox (not recommended).
3	System finetuning	Optional checks (`curl`, GD, ImageMagick, the other database extensions, TrueType text functions, `open_basedir`). Fix and press **Finetune**, or press **Next** to go on.
4	Outgoing Email	Sendmail/MTA, or SMTP with server, user and password.
5	Choose database system	SQLite is listed first and preselected (`settings/setup.ini [DatabaseSettings] DefaultType=sqlite3`). If the `sqlite3` extension is missing, the first available system is chosen and the page says so.
6	Database initialization	Server, port, database name, user, password, socket. For SQLite only the file name (default `sqlite.db`, kept in `var/storage/sqlite3/`); the Database field of Site details later offers the SQLite files already there.
7	Language support	The primary language and any additional languages.
8	Site package	The site package to install; the bundled one is preselected.
9	Package language options	Shown when the package has languages you did not choose: create each as a language of its own (preselected), map it to one of yours, or skip its content.
10	Site access configuration	How the siteaccesses are told apart: by URL path, by port, or by host name.
11	Site details	Title, URL, the match values of `site`, `admin` and `editor` (URL paths are prefilled with the package identifier, *identifier*`_admin` and `editor`; ports `8080`, `8081`, `8082`; host names *identifier*`.`*host*, *identifier*`-admin.`*host*, `edit.`*host*). The wizard refuses `admin` and `user` as values. Also the sender of the site's optional e-mail, and what to do with a database that already holds data.
12	Site administrator	First name, last name, e-mail and password of the `admin` user. The password must be at least 10 characters (`site.ini [UserSettings] MinPasswordLength`); an empty one is refused.
13	Site security	Advice when the site does not run in virtual host mode: copy the rules with `./console exp:setup:htaccess --apply` (or `cp .htaccess_root .htaccess`).
14	Registration	Nothing to enter: a page about the community. The wizard sends no registration e-mail.
—	Create sites	Not a page: the database schema and data, the packages and the `settings/siteaccess/` files are written here.

#	Page	What to enter
15	Finished	The addresses of the public, admin and editor sites.

The wizard logs its progress to `var/log/setup.log`. More: setup wizard and editor siteaccess, clean install defaults, installer logs and seed data.

6. Install with the kickstarter

Full chapter: 06-kickstarter

The kickstarter (`bin/php/kickstarter.php`, console `exp:kickstarter`) runs the setup wizard's steps without a browser and reads every answer from `kickstart.ini` in the installation root. The commented template is `kickstart.ini-dist`.

The commands

Shell

```
php bin/php/console exp:kickstarter ini            # write kickstart.ini in an interactive editor
php bin/php/console exp:kickstarter ini --yes      # write it with the built-in defaults, no ques
tions
php bin/php/console exp:kickstarter run --dry-run  # check kickstart.ini, the database and the pa
ckages
php bin/php/console exp:kickstarter run --apply --force    # install
```

`php bin/php/kickstarter.php` takes the same subcommands and options.

Subcommand and option	Meaning
`ini`	Write or edit `kickstart.ini` from `kickstart.ini-dist` in an interactive editor (needs a terminal).
`ini --yes, -y`	Accept the built-in defaults and write without asking, with mode `0600`: SQLite in `exponential.db`, `DatabaseAction=ignore` (installs into an empty database, drops nothing). Check the lines marked `# REVIEW` before you run it, above all `URL`, which is empty.
`ini --defaults, -d`	Copy the example values of `kickstart.ini-dist`.
`ini --from-installed`	Take the database server, name and user (never the password) of an installed siteaccess as defaults. The file then points at a live database.
`run`	Run the steps from `Welcome` to `Final`.
`run --apply`	Required whenever the steps include `CreateSites`, the step that writes the database and settings. Without it `run` stops with "Re-run with --force to confirm you want to install the site package."
`run --dry-run`	Validate `kickstart.ini`, run `DatabaseChoice` to `SiteDetails`, download the site package and its dependencies into a temporary `var/storage/packages/dryrun/` (removed again), and stop. The database must be reachable; nothing is written to it, no password is generated and no mail sent.
`run --list-steps`	List the steps and exit. Changes nothing.
`run --start-step=<step>, --stop-step=<step>`	Run part of the steps, for diagnosis. A run cannot resume an earlier one: a start step whose later steps need the results of earlier ones is refused, with the step to start at. After a failure, fix the cause and run the whole sequence again.

While `run --apply` rebuilds the database, the site answers every visitor with the maintenance page; a maintenance window that was already open is kept, and after a failed step the page stays until you run again or `php bin/php/maintenance.php off`. Every run is logged to `var/log/kickstart.log` with passwords masked (earlier runs are kept as `kickstart.log.1` to `.9`; `EXP_KICKSTART_LOG=0` turns the log off). The site package is taken from `var/storage/packages/` when it is already imported there; otherwise it is downloaded, with the packages it requires, from the package index configured in `settings/package.ini`.

kickstart.ini

One section per wizard step. Leave no whitespace before a section or a key. `Continue=true` skips the step and uses the values; `Continue=false` (or no `Continue`) shows the step pre-filled. A complete example for MySQL:

INI

```ini
[email_settings]
Continue=true
Type=mta

[database_choice]
Continue=true
Type=mysqli

[database_init]
Continue=true
Server=localhost
Port=
Database=exponential
User=exponential
Password=CHANGE_ME
Socket=

[language_options]
Continue=true
Primary=eng-GB
Languages[]=eng-US

[site_types]
Continue=true
Site_package=sevenx_multisite

[site_access]
Continue=true
Access=url

[site_details]
Continue=true
Title=My Exponential site
URL=https://www.example.com
Access=site
AdminAccess=admin
EditorAccess=editor
Database=exponential
DatabaseAction=remove

[site_admin]
Continue=true
FirstName=Site
LastName=Administrator
Email=webmaster@example.com
Password=
```

```
[security]
Continue=true

[registration]
Continue=true
Send=false
```

Section	Keys
email_settings	Type (mta or smtp), Server, User, Password
database_choice	Type: sqlite3 (default), mysqli (or mysql), pgsql (or postgresql), mongodb, oci8 (or oracle; needs extension/ezoracle)
database_init	Server, Port, Database, User, Password, Socket. For Oracle, Database is the connect string (for example 127.0.0.1:1521/FREEPDB1) or a TNS alias.
language_options	Primary, Languages[]
site_types	Site_package
site_access	Access: url, port or hostname
site_details	Title, URL, OrganisationName, OrganisationAddress, Access, AdminAccess, EditorAccess, AccessPort, AdminAccessPort, EditorAccessPort, AccessHostname, AdminAccessHostname, EditorAccessHostname, Database, DatabaseAction
site_admin	FirstName, LastName, Email, Password (empty: one is generated and written to var/log/initial-admin-password)
security	Continue only
registration	Send (default false; true mails a report only to settings/setup.ini [RegistrationSettings] Receiver, which is empty by default, so nothing is sent), UserData[...]

Every language must have a locale in `share/locale`, or the run stops in `LanguageOptions` and names the language. Languages of the site package that you did not list are created as languages of their own, so their content stays reachable.

DatabaseAction: read this before you run

> **Warning.** `DatabaseAction=remove` **empties the named database** and installs into it. Everything in that database is lost. Point `Database` at a database that holds nothing you need, and take a backup first.

Value	What happens to the database
remove	Existing entries are removed, then the schema, data and packages are installed. A clean install.
ignore	Entries are added without cleaning up.
skip	No schema or data is inserted. Use it to regenerate the siteaccess and INI files against an existing database.

Re-running

- Run again with the same `kickstart.ini` for an identical install. With `DatabaseAction=remove` this replaces the database again.
- After a failure, fix the cause (the failing step and its message are in the output and in `var/log/setup.log`) and run the whole sequence again; with `DatabaseAction=remove` the second run replaces what the first left.
- `ini` and `run` delete the cached `var/cache/ini/kickstart-*.php` themselves. If you edit `kickstart.ini` by hand and see old values, delete those files.
- When you install onto a different database engine than the one the installation used before, first set `DatabaseImplementation` in `settings/override/site.ini.append.php` to the new engine: the installer keeps the value it finds there.
- `kickstart.ini` holds the database password. Keep it out of version control, and remove or move it after the install if the web wizard must not read it.

Full reference: Kickstarter CLI, Kickstarter.

7. Install with one console command

Full chapter: 07-console-install

`exp:install` (`bin/php/install.php`) builds the kickstart configuration from its options, runs the same steps in the same process, and puts back any `kickstart.ini` that was there. Everything has a default, so this alone installs the `sevenx_multisite` package on SQLite with the user `admin` and a generated password:

Shell

```
php bin/php/console exp:install --apply
```

Other databases (keep the password out of the shell history with `EXP_INSTALL_DB_PASSWORD`):

Shell

```
EXP_INSTALL_DB_PASSWORD='secret' php bin/php/console exp:install --apply --db=mysql --db-host=127
.0.0.1 \
    --db-name=mysite --db-user=mysite --url=https://www.example.com --random-password
```

Option	Default	Meaning
`--db=`	`sqlite`	`sqlite`, `mysql` (also `mariadb`), `pgsql`, `mongodb`, `oracle`
`--db-host=`, `--db-port=`, `--db-socket=`	`localhost`; 3306, 5432, 27017, 1521	where the database server is
`--db-name=`	`exponential` (`exponential.db` for SQLite, FREEPDB1 for Oracle)	the database
`--db-user=`, `--db-password=`	`root` (mysql), `postgres` (pgsql), none otherwise	credentials; or `EXP_INSTALL_DB_PASSWORD`
`--db-action=`	`remove`	`remove`, `ignore` or `skip`, as `DatabaseAction` above. `remove` **empties the database first.**

Option	Default	Meaning
`--package=`	`sevenx_multisite`	site package
`--language=`, `--languages=`	`eng-US`	primary language; more as a comma list
`--title=`	`Exponential`	site name
`--url=`	`http://localhost` (`http://<--host>` with `--access=host`)	where the site is; give the real address, it ends up in every link the site writes into mails
`--access=`	`url`	`url`, `host` or `port`
`--site-access=`, `--admin-access=`, `--editor-access=`	`site`, `admin`, `editor`	the URL paths of the three siteaccesses (their names stay `site`, `admin`, `editor`)
`--host=`, `--admin-host=`, `--editor-host=`	—, —, `edit.<--host without www.>`	host names (imply `--access=host`; `--host` and `--admin-host` are both required)
`--port=`, `--admin-port=`, `--editor-port=`	`8080`, `8081`, `--port` + 2	ports for `--access=port`
`--organisation-name=`, `--organisation-address=`	the site name, empty	the sender of the site's optional e-mail and its postal address
`--email=`, `--first-name=`, `--last-name=`		the administrator
`--password=`	generated	kept only when it has at least 10 characters (`site.ini [UserSettings] MinPasswordLength`) and is not a well-known one; otherwise replaced by a generated one
`--random-password`		generate a 24-character password, shown once and written to `var/log/initial-admin-password`
`--force`		install over an existing installation (replaces its database and settings)
`--dry-run`		check the configuration and packages, install nothing
`--print`		show the configuration that would be used, and stop
`--help`		all options

The summary at the end lists the site, admin login and editor addresses, the user and the password. The configuration used is kept in `var/log/exp-install-<date>.ini` with passwords masked. More: Install in one command.

8. Serve the site: web server, HTTPS and permissions

Full chapters: 08-serving-the-site and 13-security-hardening

The document root is always the installation root.

Option A: Exponential Velocity (recommended)

Velocity is an application server that ships for Exponential. It loads the application once, keeps it in persistent workers and has its own response cache. It serves HTTP and HTTPS itself, with its own certificates, so no Apache, nginx or PHP-FPM is needed. Install it with Composer (section 3), then:

Shell

```
php bin/php/console exp:velocity config set --apply ServerSettings Engine qbix
php bin/php/console exp:velocity start
php bin/php/console exp:velocity status
```

`config set` writes `settings/override/velocity.ini.append.php`; the shipped defaults stay in `settings/velocity.ini`. The shipped default engine is `php` (development); set `qbix` for every real site. The `qbix` engine serves plain HTTP on `[ServerSettings] Port` (8088) and HTTPS on `HTTPSPort` (8080), bound to `Host` (`127.0.0.1`). To serve the network:

Shell

```
php bin/php/console exp:velocity config set --apply ServerSettings Host 0.0.0.0
php bin/php/console exp:velocity config set --apply ServerSettings Port 80
php bin/php/console exp:velocity config set --apply ServerSettings HTTPSPort 443
php bin/php/console exp:velocity restart
```

Ports below 1024 need root. Started as root, the server process keeps root to bind the ports and read the certificate, and the workers run as `[ServerSettings] User` and `Group` (when empty: `VC_RUN_USER` and `VC_RUN_GROUP` from the environment or `/etc/vc/envvars`, else the owner of the document root). Workers never run as root unless `AllowRootWorkers=enabled`.

HTTPS with a certificate you have (one PEM file may hold both):

Shell

```
php bin/php/console exp:velocity config set --apply HTTPSSettings Certificate /etc/ssl/example.co
m/fullchain.pem
php bin/php/console exp:velocity config set --apply HTTPSSettings Key /etc/ssl/example.com/privke
y.pem
php bin/php/console exp:velocity config set --apply HTTPSSettings Enabled true
php bin/php/console exp:velocity restart
php bin/php/console exp:velocity ssl show
```

HTTPS is on when `Enabled` is `true` and both files exist. The certificate is checked before use (parses, not expired, key matches); a self-signed certificate stands in while the real one is unusable (`fallback`, default `self-signed`), and a renewed certificate is swapped in without a restart. The engine's certificate subsystem also obtains and renews Let's Encrypt (or any ACME CA) certificates and reads archives and PKCS#12 bundles; it is configured under `Q.web.https` and described in Velocity HTTPS and certificates. `[HTTPSSettings] HSTSMaxAge` (shipped as 0, off; 300 to start) sets Strict-Transport-Security; raise it only when every name of the site is on HTTPS for good.

Configuration layout. Velocity's configuration follows Debian's `/etc/apache2`: `/etc/vc/vc.conf`, `ports.conf`, `envvars`, and `sites-available/sites-enabled`, `conf-available/conf-enabled`, `mods-available/mods-enabled` (an overlay on the engine's `/etc/qbix`, which is read too). The site file `sites-available/<site>.conf` is generated from `velocity.ini` on every `start` and `restart`, or with `exp:velocity layout migrate`. Switch sites, snippets and modules like `a2ensite`:

Shell

```
php bin/php/console exp:velocity layout                # show every file the server uses
php bin/php/console exp:velocity site enable --apply <name>
php bin/php/console exp:velocity site disable --apply <name>
```

Day to day:

Command	Does
`exp:velocity start`, `stop`, `status`, `restart`	as named; `status --json` for monitoring
`exp:velocity graceful`	re-executes the server, keeping the listening socket: no dropped requests
`exp:velocity cache clear`	clears the response cache
`exp:velocity deploy`	after a PHP, INI or class change: autoloads, caches, PHP-FPM reload, Velocity restart, in the right order (`--dry-run` shows the steps)

Workers keep the application in memory: after changing a class or a setting, restart (or `deploy`); clearing caches alone does not reach them. Velocity has no command to install a boot service; the engine package ships an example systemd unit in `vendor/se7enxweb/exponential-velocity/service/`. When Velocity serves the site alone, set `[DeploySettings] PhpFpmService` to `disabled`.

More: Velocity engines, on-disk layout, persistent workers, Deploying.

FrankenPHP is the production-ready alternative engine driven by the same commands: `exp:velocity config set ServerSettings Engine frankenphp`, `exp:velocity install` (downloads and checks the binary), `exp:velocity start`. It serves HTTPS by default (`[FrankenPHPSettings] HTTPS=enabled`, port 8444), with your `[HTTPSSettings] Certificate` and `Key`, or a self-signed certificate it makes and renews itself.

Option B: Apache with PHP-FPM

Enable the shipped rewrite rules:

Shell

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

The command copies `.htaccess_root` to `.htaccess` with the template's mode (and, run as root with `--allow-root-user`, its owner), shows a diff and refuses when a `.htaccess` that differs is already there, and backs up the old file before `--force` `--apply` replaces it. `--static` copies `.htaccess_root_static` instead. The same way, `./console exp:setup:config --apply` puts `config.php-RECOMMENDED` into place as `config.php`; see Setting up .htaccess and config.php from the console.

They turn 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 remove security rules from it to make something work. A minimal virtual host:

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>
```

`mod_rewrite`, `mod_proxy` and `mod_proxy_fcgi` must be loaded. Check with `apachectl configtest` and reload Apache. For HTTPS add a `*:443` virtual host with `SSLEngine on`, `SSLCertificateFile` and `SSLCertificateKeyFile`, and a certificate from your usual tool. After a PHP change reload the PHP-FPM that serves the site; on servers with several PHP versions that is not always the one called `php-fpm` (`[DeploySettings] PhpFpmService=auto` lets `exp:velocity deploy` find it). Details: Deploying.

Permissions

The user the web server or Velocity workers run as must be able to write these directories (the setup check `directory_permissions` in `settings/setup.ini`): `design`, `extension`, `settings`, `settings/override`, `settings/siteaccess`, `var` and everything below it that Exponential writes (`var/cache`, `var/log`, `var/storage`, `var/autoload`). The system check prints the commands for your machine, of this form:

Shell

```
sudo chmod -R ug+rwx design extension settings var
sudo chown -R <web user>:<web group> design extension settings var
```

When command-line scripts and cronjobs run as a different user from the web server, give them a common group that can write the same files.

Behind a reverse proxy or load balancer

Forwarded headers (`X-Forwarded-For`, `X-Forwarded-Proto`, `X-Forwarded-Host`) are believed only from the addresses in `site.ini [HTTPHeaderSettings] TrustedProxies[]`, by default `127.0.0.1` and `::1`. A proxy on another machine must be added there (for example `TrustedProxies[]=10.0.0.0/8` in `settings/override/site.ini.append.php`), or the site builds `http://` links behind an HTTPS proxy and, with `ClientIpByCustomHTTPHeader=X-Forwarded-For`, sees the proxy's address instead of the visitor's. Details: chapter 8 and Forwarded headers are trusted only from configured proxies.

9. SQLite in production

Full chapter: 09-databases

SQLite is not only for trying Exponential. The kernel driver (`lib/ezdb/classes/ezsqlite3db.php`, class `eZSQLite3DB`) is built for a site under load.

What the driver does:

WAL	The driver switches the database to `journal_mode=WAL` (set once, stored in the file). Readers do not block the writer and the writer does not block readers, so reads run concurrently.
Connection tuning	`synchronous=NORMAL` (crash-safe with WAL), `cache_size=-65536` (64 MB), `mmap_size=268435456` (256 MB), `temp_store=MEMORY`, `busy_timeout` 5000 ms in a web request (`SQLiteBusyTimeout`) and 30000 ms on the command line (`SQLiteCLIBusyTimeout`).
Queued writes	SQLite has one writer at a time. Each transaction starts with `BEGIN IMMEDIATE` after taking an exclusive lock on `<database file>.writer-lock`, next to the database. Writers wait their turn there for up to `SQLiteTransactionWait` seconds instead of failing with "database is locked". Waiting is safe: nothing has been written yet, and a started transaction cannot fail for a lock. The lock is released at commit or rollback, and by the operating system if the process dies.
Single writes	A write outside a transaction queues on the same lock file and waits up to the statement wait; a lock SQLite reports at once is tried again within it.
Command line	Cronjobs and scripts wait `SQLiteCLITransactionWait` seconds (default 300) for a transaction and `SQLiteCLIBusyTimeout` ms (default 30000) for a statement, so a cronjob waits for a long import instead of failing.
Timeout	Only work that waits longer than its limit fails, with "database is busy: the transaction could not start within N s (waited N s, [DatabaseSettings] <setting>) ..." or "database is busy: the statement waited N s for another write to end (limit N s, [DatabaseSettings] <setting>) and was not run; nothing was written".

Settings (`settings/site.ini`, block `[DatabaseSettings]`; override in `settings/override/site.ini.append.php`):

INI

```ini
[DatabaseSettings]
DatabaseImplementation=sqlite3
# a name under var/storage/sqlite3/, or an absolute path
Database=exponential.db
# seconds a transaction waits for the writers ahead of it (default 25);
# keep it below the request timeout: Velocity's Q.webserver.requestTimeout
# (30 s unless changed), or PHP-FPM's request_terminate_timeout
SQLiteTransactionWait=25
# the same on the command line (cronjobs, imports), seconds
SQLiteCLITransactionWait=300
# ms a single statement waits for another write: web requests, command line
SQLiteBusyTimeout=5000
SQLiteCLIBusyTimeout=30000
# PRAGMAs over the driver's defaults, one per line as name=value
SQLitePragmas[]
#SQLitePragmas[]=cache_size=-131072
```

Clear the INI cache after a change: `php bin/php/ezcache.php --clear-tag=ini`.

What carries the load. Most requests never reach the database: the content view cache, the template-block cache and, with Velocity, the response cache, which answers a repeat request without starting the application. Under Velocity's persistent workers every worker keeps the application loaded and shares the one database file; reads run side by side and writes queue on the writer lock (the driver reopens the lock in each forked worker). In load tests of this driver, concurrent readers scaled with the number of processes, short writes stayed error-free at 16 concurrent writers at the serial rate, and long transactions that failed with the earlier driver's 5-second limit completed without errors, only waiting longer. A team where many editors publish large changes at the same moment is still better served by MySQL, MariaDB or PostgreSQL.

Ownership and permissions. SQLite needs to write the database file, the `-wal` and `-shm` files beside it and the directory that holds them; the driver creates `var/storage/sqlite3/` with mode `0775` but does not set the mode or owner of the files, so they get the creating process's umask. When Apache/PHP-FPM, Velocity workers and cronjobs run as different users, give them one group that can read and write `var/storage/sqlite3/` and every file in it. A `.writer-lock` created by another user still works: the driver opens it read-only when it cannot open it for writing. Never serve the database file over HTTP; check with `curl -sI https://www.example.com/var/storage/sqlite3/exponential.db` (anything but 200 is right).

Backups. Use SQLite's online backup, never a plain copy of a busy file:

Shell

```
sqlite3 var/storage/sqlite3/exponential.db ".backup backup.db"
```

Restore by copying `backup.db` back with the site in maintenance mode (`php bin/php/maintenance.php on`, then `off`). Run `ANALYZE` once after an install or a big import so the query planner has statistics (`sqlite3 var/storage/sqlite3/exponential.db ANALYZE`).

More: SQLite database support, SQLite 3 driver, SQLite transactions.

10. After installing

Full chapter: 10-after-installing

1. **Log in** at the admin address the installer printed (for example `/admin/user/login`) as `admin`. Change the password (**Change password** in the user menu) and delete `var/log/initial-admin-password`. A lost password is reset as root with `php bin/php/resetuserpassword.php --allow-root-user -u admin -g` (without root, authorize with another administrator: `-a <login> -ap <password>`).
2. **Check the front page and the logs.** Open the site address. On an error read `var/log/error.log` and Repairing an installation.
3. **Set up the cronjobs.** They publish scheduled content, send notifications and newsletters and clean up. List the parts, then add them to the crontab of the site user:

   Shell

   ```
   php runcronjobs.php --list
   ```

   ```
   */5 * * * * cd /path/to/installation && php runcronjobs.php --siteaccess=site frequent >> var
   /log/cron-frequent.log 2>&1
   17 * * * *  cd /path/to/installation && php runcronjobs.php --siteaccess=site infrequent >> v
   ar/log/cron-infrequent.log 2>&1
   ```

   Use your public siteaccess name. The same parts are in the admin under **Setup > Cronjobs**, and Velocity's scheduler can run them without the system cron.
4. **Know the cache commands.** Clear the smallest thing that fixes the problem:

You changed	Command
a setting (`.ini`)	`php bin/php/ezcache.php --clear-tag=ini`
a template (`.tpl`)	`php bin/php/ezcache.php --clear-tag=template`

You changed	Command
content looks stale	`php bin/php/ezcache.php --clear-tag=content`
added a template or an extension	`php bin/php/ezcache.php --clear-id=template-override`
unsure	`php bin/php/ezcache.php --clear-all`

With Velocity, also `php bin/php/console exp:velocity cache clear --apply`; after a class or INI change, `exp:velocity deploy` does all of it in order.

5. **Regenerate autoloads** after adding or renaming a class in an extension: `php bin/php/ezpgenerateautoloads.php -e` (`-k` for the kernel).
6. **Back up** the database and `var/storage`, `settings/override`, `settings/siteaccess` (Operating a site).
7. **The help's manuals.** The kickstarter and `exp:install` have queued the build of the help's manuals from the Markdown help (the last lines of their output): a background job makes them over the next 20 to 30 minutes, and `./console exp:help:content report -s admin` says "Manuals are being built (n of 52)" until it is done. After the setup wizard, or when queueing reported a problem, run `./console exp:help:content build-manuals --apply -s admin --queue`. Run `./console exp:help:content build-manuals --apply -s admin --release` after each upgrade (Manuals from the Markdown help).
8. **Optional: the job queue and Exponential Intelligence.** `composer.json` brings the job queue `expqueue` and the writing assistant `expai` into `extension/`, both inactive. To use them:

INI

```
# settings/override/site.ini.append.php: the job queue, for every siteaccess
[ExtensionSettings]
ActiveExtensions[]=expqueue
```

INI

```
# settings/siteaccess/<each admin siteaccess>/site.ini.append.php: the assistant, for the adm
inistration only
[ExtensionSettings]
ActiveAccessExtensions[]=expai
```

Shell

```
./console ext:expqueue:install-schema
php extension/expai/bin/php/expaischema.php --install -s admin
php bin/php/ezpgenerateautoloads.php -e
php bin/php/console exp:velocity deploy --packer    # or clear the caches and reload PHP-FPM
```

Put the provider key into `settings/override/expai.ini.append.php` (`[AISettings] Enabled=true`, `ApiKey=...`; never into version control), give editors the policy `expai/write`, and add two cron lines, both with the admin siteaccess because expai is active there only:

```
* * * * *   cd /path/to/installation && php runcronjobs.php -q -s admin expqueue
*/5 * * * * cd /path/to/installation && php runcronjobs.php -q -s admin expai
```

What they do and every further step: Exponential Intelligence and the job queue.

Then continue with the guides:

- Configuring Exponential 6.0: where settings live and which file wins, siteaccesses, languages, mail, caches, security, extensions and logs.
- Extending Exponential 6.0: extensions, modules and views, templates, datatypes, workflow events, cronjob parts, commands, tests and releases.
- Integrating With Exponential 6.0: the APIs, machine sign-in, content and events in and out, single sign-on, front ends and security for integrations.
- Distributions of Exponential: every distribution of Exponential, its PHP versions and databases, and which one to choose.
- Getting started: first login, first content, first template change.
- The content model and editing content: classes, objects, the online editor, the trash and content jobs.
- Templates and design: find the template that wrote a page, override it, switch the admin design.
- Extensions: find, install, configure, build and release an extension.
- Remote services and apps: call Exponential from the shell, from Python and with an API token.
- Deploying: serve the site with Apache and PHP-FPM, Velocity or FrankenPHP, over HTTPS.
- Operating a site: caches, cronjobs, backups, logs and repairs.
- Security and audit: check the hardening, roles and policies, the audit trail.
- Every guide: the learning path.

11. Upgrading from an older version

Full chapter: 11-upgrading

Do not run the installer over an existing site. Follow Upgrading: from 4.x, 5.x or an earlier 6.0.x to the current line, with the database update and the checklists per release. A 6.0 installation is brought up to date with the short procedure of Updating Exponential 6.0. After any upgrade, compare `.htaccess` and `config.php` with the new templates: `./console exp:setup:status`, `./console exp:setup:htaccess` (the diff) and `./console exp:config:list` (settings new in `config.php-RECOMMENDED` show up as `(not set)`). To move a site elsewhere or remove it, export everything first with Uninstalling Exponential 6.0.

The online manual has further installation notes:
https://exponential.doc.exponential.earth/Exponential/Technical-manual/6.x/Installation.html

12. Troubleshooting

Full chapter: 12-troubleshooting

When the table does not help, Getting Support shows where to ask, from the communities of PHP and the database to the Exponential community and 7x.

Symptom	Cause and fix
Every address shows the setup wizard	`CheckValidity` is still `true`: the install did not finish. Finish the wizard, or run the kickstarter or `exp:install`.

Symptom	Cause and fix
"The site is being set up" (503)	A wizard is running in another browser (the hold ends with its last page or 30 minutes after its last request), or a kickstarter run is rebuilding the database. If nothing is running: `php bin/php/maintenance.php status`, then `off`.
The wizard skips pages or shows values you did not enter	A `kickstart.ini` is in the installation root. Move it away.
The system check fails on directory permissions	Run the chmod/chown commands the page prints (Permissions), then **Next**.
`The CreateSites step will modify the database and site settings`	The kickstarter needs `--force` to install.
`exp:install`: "This directory already holds an installation"	Add `--force` only if you mean to replace it; `--dry-run` checks without changing anything.
`kickstart.ini` values are ignored	Leading whitespace before sections or keys, a missing `Continue=true`, or a stale `var/cache/ini/kickstart-*.php`.
After switching database engine no page loads	`DatabaseImplementation` in `settings/override/site.ini.append.php` still names the old engine. Set it to the new one.
Kickstarter: "The database file name is not valid"	A `kickstart.ini` written by `ini --yes` before 5 October 2026 has `Database=ezp`, and SQLite needs a name ending in `.db`, `.db3`, `.sqlite` or `.sqlite3`. Set `[database_init] Database` and `[site_details] Database` to, for example, `exponential.db`, and `DatabaseAction` to `ignore` for a new database.
Kickstarter: "--start-step=... cannot work"	A run cannot resume an earlier one. Fix the cause and run from `Welcome` (the default).
SQLite: "database is busy: the transaction could not start within N s"	Another write held the lock for the whole `SQLiteTransactionWait`. Nothing was written; retry, or raise the setting (below the request timeout).
404 for every page under Apache	`.htaccess` is not read (`AllowOverride All`) or `mod_rewrite` is off; is there a `.htaccess` at all? `./console exp:setup:status` says, `./console exp:setup:htaccess --apply` copies it.
`FAIL: .htaccess exists and differs ...` or `FAIL: config.php exists and differs ...`	The file has changes of its own. Read the diff (`-v` shows all of it); `--force --apply` replaces it after a backup to `var/backup/setup/<time>/`. See Setting up .htaccess and config.php from the console.
`Warning: config.php does not compile`	Fix the line it names, or `./console exp:setup:config --force --apply` (the broken file is backed up first).
A white page or 500	Read `var/log/error.log`; check `php -v` and that `vendor/` exists (`composer install`).
A change does not show under Velocity	Workers keep the old code: `exp:velocity restart` (or `deploy`), then `exp:velocity cache clear`.
The administrator password is lost	As root: `php bin/php/resetuserpassword.php --allow-root-user -u admin -g`; otherwise authorize with another administrator (`-a <login> -ap <password>`). See Reset a user password.

More: Repairing an installation, Maintenance mode.

Public demos: demo mode

A public demo whose administrator sign-in is published runs with the environment variable `EXP_DEMO_MODE=1`. Every installer (setup wizard, kickstarter, `exp:install`) then installs the published administrator password as given, also a well-known or short one such as `publish` or `publishing` (never an empty one), and afterwards nobody can change or reset an administrator password while the variable is set. Never set it for a real site. See Demo mode and the demo mode examples.

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.