Application servers

The three ways Exponential is served - a web server with PHP-FPM, Exponential Velocity with persistent workers and a response cache, and FrankenPHP in classic mode - compared by process model, state between requests...

Source file: `doc/eng-us/help/software-engineers/application-servers.md` Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/software-engineers/application-servers.md

This page is for engineers who choose how Exponential is served, or who need to know which runtime their code will meet. Since 6.0.15 Exponential ships its own application server, Exponential Velocity, and drives three engines with the same command and settings file. The source of truth is Velocity engines.

Contents

1. The options
2. Compared
3. What the choice means for code
4. Driving the engines
5. Choosing

The options

- **A web server with PHP-FPM** (Apache, nginx and the other servers the README lists). Every request starts PHP from a clean state. Exponential runs behind any web server that can rewrite a URL to `index.php`.
- **Exponential Velocity**, engine `qbix`. A web server written in PHP: the parent loads the application once and forks persistent workers that serve many requests and are reset after each; a response cache can answer repeat public requests before a worker is used. As of 6.0.15 it is required by `composer.json` (`se7enxweb/exponential-velocity`), so every installation has it. The project calls it the recommended engine for every stage (Velocity is part of Exponential 6.0.15).
- **FrankenPHP**, engine `frankenphp`. Caddy with PHP built in, one binary, downloaded by `exp:velocity install` and checked against a SHA-256 pin. PHP runs in classic mode (a clean state per request); worker mode is deliberately not offered (FrankenPHP).
- **PHP's built-in server**, engine `php`: the shipped default in `velocity.ini`, because it needs nothing installed. For development only.

Compared

	PHP-FPM	Velocity (`qbix`)	FrankenPHP	`php -S`
Role in the project	traditional, fully supported	recommended, every stage	production-ready alternative	development only

	PHP-FPM	Velocity (`qbix`)	FrankenPHP	`php -S`
State between requests	none	a persistent worker, reset after each request	none (classic mode)	none
Response cache	no	yes (`[CacheSettings]`)	not yet	no
TLS and HTTP/2	your web server's	yes	yes	no
Graceful reload	`systemctl reload` of the FPM service	re-exec, socket kept	configuration reload	restart
A class change is live after	reloading FPM (OPcache)	`exp:velocity restart` or `deploy`	its reload	restart

What the choice means for code

The same code runs under all of them, and the kernel's rule is that every change **must** keep working under PHP-FPM, Velocity and FrankenPHP. What the persistent model adds:

- static properties, globals and registrations survive from one request to the next unless they are reset (Concurrency, statics and long-lived processes);
- `exit` and `die` end the request, not the worker, through a compatibility layer; kernel code uses `eZExecution::cleanExit()`;
- a worker started before a deploy does not know a new class until it is restarted.

Driving the engines

Shell

```
php bin/php/console exp:velocity status                          # every engine: role, state, URL
php bin/php/console exp:velocity start                           # the default engine
php bin/php/console exp:velocity graceful                        # reload workers one at a time
php bin/php/console exp:velocity deploy --dry-run --allow-root-user
```

`settings/velocity.ini [ServerSettings] Engine` chooses the default engine (`php` shipped, `qbix` recommended, `frankenphp`). The verbs are the same for every engine: `start`, `stop`, `graceful`, `restart`, `kill`, `status`, `deploy`, `command`, `config`, `install`.

Choosing

Situation	Choice
A new site, any stage	Velocity, as the project recommends
A host that only offers PHP-FPM	PHP-FPM behind its web server; everything works, without the response cache
One verified binary with TLS and HTTP/2, no persistent state	FrankenPHP
Code you do not trust in a long-lived process yet	PHP-FPM or FrankenPHP until it is reviewed, or Velocity's `forkPerRequest` mode

Measure on your own pages before you decide (Performance measurement and benchmarks).

Related pages

- Velocity persistent workers
- Velocity: running Exponential in a persistent-worker web server
- Deploying with Velocity
- 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.