The compiled kernel container

What var/cache/kernelservices/ is, what it saves on every request under PHP-FPM and Velocity, when the deploy and the INI cache clear rebuild it, how to check it with exp:services:check, and how to switch it off.

Source file: `doc/eng-us/help/hosting/performance/compiled-kernel-container.md` Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/hosting/performance/compiled-kernel-container.md

This page is for system administrators who deploy and run Exponential sites. As of 6.0.15 the kernel keeps a **compiled container** in `var/cache/kernelservices/`: for every siteaccess, a recording of how the kernel activates the extensions at the start of each request. With it, a request no longer looks up every extension directory on disk, and on Exponential Velocity a worker builds the settings files of a siteaccess once instead of on every request. The developer side is on the kernel service container; the full reference on kernel services.

What it saves

Measured on a test installation with 62 active extensions and 18 siteaccesses (the start of a request, up to the module run):

	Before	After
PHP-FPM, start of a request	38 to 50 ms	27 to 36 ms (-15 to -35%)
Velocity, start of a request	21 to 31 ms	7 to 14 ms (-51 to -72%)

Rendering is not changed by it; a page served from the HTTP cache never starts the kernel and gains nothing.

Nothing to do, normally

When	What happens
`exp:velocity deploy`	the INI step (`ezcache.php --clear-tag=ini`) compiles it again; Velocity's workers notice the new one on their next request
`php bin/php/ezcache.php --clear-tag=ini` or `--clear-all` on the command line	removed and compiled again (a second or two)
INI cache cleared in the administration	removed; each siteaccess records itself again on its next web request
First request after an install	the siteaccess records itself; the next request uses it

A request uses the compiled container only when it starts from exactly the state that was recorded; otherwise it works as before. A stale container therefore makes a request slower, never different.

Checking it

Shell

```
./console exp:services:check      # PASS or FAIL, the siteaccesses compiled, the generation, the
worker scope
./console exp:services:compile    # compile it now
```

`exp:services:check` exits with 1 on an error, so it can run in a deploy check or a monitor.

Settings

`settings/override/kernelservices.ini.append.php`:

INI

```
<?php /* #?ini charset="utf-8"?

[ContainerSettings]
# compiled (default), dev (nothing compiled is used) or disabled (the container takes no part in
the boot)
Mode=compiled
# a web request records a siteaccess that has no current compiled file (default enabled)
AutoRebuild=enabled
# keep each siteaccess's settings files for a Velocity worker: auto = when config.php sets EZP_IN
I_FILEMTIME_CHECK
# to false (the INI cache does not look at file times either)
WorkerScope=auto
*/ ?>
```

To switch it off, set `Mode=disabled` and clear the INI cache: the kernel boots exactly as before 6.0.15.

Things to know

- **Ownership.** Velocity runs as root, PHP-FPM as the site's user. A file the container writes as root is given to the owner of `var/`, so the other can replace it.
- **Editing a settings file by hand.** As for the INI cache: with `EZP_INI_FILEMTIME_CHECK` false in `config.php`, nothing notices the edit until the INI cache is cleared (or the deploy runs), and that also rebuilds the container.
- **A new extension directory** that appears on disk without a settings change needs a deploy or `exp:services:compile`, as it needs new autoloads.
- **Several sites on one installation** (multi-site hosting, host matching): one compiled file per siteaccess, and a Velocity worker keeps the settings of each siteaccess apart.

Learn more

- Kernel services (reference)
- Cache clearing and warming
- Velocity workers and pools
- Deploying changes

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.