Environments and secrets: EXP_ENV and settings/override

The Exponential counterpart of APP_ENV, .env files, Rails credentials and Spring profiles - the EXP_ENV constant in config.env.php, settings variants per environment, what the environment name changes, where secrets...

Source file: `doc/eng-us/help/web-framework-developers/environments-and-secrets.md` Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/web-framework-developers/environments-and-secrets.md

This page is for developers who switch environments with `APP_ENV`, keep secrets in `.env.local`, Rails credentials or a vault, and expect `dev`, `test` and `prod` to behave differently. Exponential names the environment of a machine with one constant, `EXP_ENV`, and lets every settings file have a variant for it (Settings per environment (EXP_ENV)).

Contents

1. Naming the environment
2. Settings variants
3. What the name changes
4. Secrets
5. Environment variables
6. Compared with other frameworks

Naming the environment

Copy `config.env.php.example` to `config.env.php` in the root of the installation:

PHP

```php
<?php
define( 'EXP_ENV', 'test' );
```

`config.env.php` is read before `config.php` by the web entry points and the command line scripts, and the shipped `.gitignore` keeps it out of version control. A name is lowercase letters, digits, `-` or `_`, starts with a letter and has at most 32 characters; an invalid name is treated as no environment.

Settings variants

The environment name goes right after `.ini`:

Standard file	**Variant for** `test`
`settings/site.ini`	`settings/site.ini.test`
`settings/override/site.ini.append.php`	`settings/override/site.ini.test.append.php`

Standard file	Variant for `test`
`settings/siteaccess/admin/site.ini.append.php`	`settings/siteaccess/admin/site.ini.test.append.php`
`extension/site_example/settings/site.ini.append.php`	`extension/site_example/settings/site.ini.test.append.php`

A variant is read right after its standard file in every layer and overrides that layer only, so a siteaccess setting still wins over a base variant. Each environment has its own INI cache files. Typical variants: the mail transport (`file` outside production), debug output, hosts, the Solr address.

INI

```
# settings/override/site.ini.test.append.php
<?php /*
[MailSettings]
Transport=file

[DebugSettings]
DebugOutput=enabled
*/ ?>
```

What the name changes

`EXP_ENV`	**Effect**
`dev`, `test`	`index.php` shows PHP errors in the browser (`display_errors` on)
`prod`, any other name, none	`display_errors` stays as php.ini sets it

Templates read the name with `{exp_environment()}`; PHP with `eZINI::environment()` (the name or `false`). The admin designs admin4 and admin4l can show the environment as a badge (`site.ini [AdminEnvironment] Level=auto`). Unlike Symfony's `dev` environment or Django's `DEBUG`, the name does not switch on a profiler or debug toolbar by itself: you switch those in the environment's variant file.

Secrets

Secret	Where
Database password	`settings/override/site.ini.append.php` of the installation
Your application's API keys	`settings/override/<yourfile>.ini.append.php`, mode 0640
Form token secret	`site.ini [HTMLForms] Secret` in `settings/override/`

Never in your extension's repository, a `.ini` file the web server might show, templates, job payloads, logs, URLs or commit messages. Ship each key as an empty value with a comment in your extension's INI file, and fail with a clear message when it is empty (Environment settings and secrets).

PHP

```php
$key = eZINI::instance( 'expdemo.ini' )->variable( 'ArchiveSettings', 'ApiKey' );
if ( $key === '' )
    throw new RuntimeException( 'expdemo.ini [ArchiveSettings] ApiKey is not set; put it in setti
ngs/override/expdemo.ini.append.php' );
```

Environment variables

Settings are not read from environment variables the way `%env(...)%`, `env()` or `os.environ` read them. A few environment variables do exist for specific purposes, for example `EXP_DEMO_MODE=1` for public demos (Environment variables). If your platform injects secrets as environment variables, read them in your own code with `getenv()` and document it, or have your deployment write them into `settings/override/`.

Compared with other frameworks

Framework	Environment	Secrets	Exponential
Symfony	`APP_ENV`, `when@env`	`.env.local`, secrets vault	`EXP_ENV`, variants; `settings/override/`
Laravel	`APP_ENV`	`.env`	the same
Django	settings modules, `DJANGO_SETTINGS_MODULE`	environment, vaults	the same
Rails	`RAILS_ENV`	encrypted credentials	the same; no encryption built in
Spring	profiles	property sources, vaults	the same

Related pages

- Configuration: INI files
- Settings per environment (EXP_ENV)
- Security standard
- 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.