
Studio 24 WordPress Multi-Environment Config

This repository contains Studio 24's standard config setup for WordPress, which loads different config based on current environment. This allows you to have different site configuration (e.g. debug mode) for different environments (e.g. production and staging).

Credit is due to FocusLabs EE Master Config who gave me the inspiration for the organisation of the config files.

Please note the current version is v2, if you need to use the older v1 version please see the v1 release.

Contributing

Strata is an Open Source project. Find out more about how to contribute.

Security Issues

If you discover a security vulnerability within WordPress Multi-Environment Config, please follow our disclosure procedure.

How it works

The system detects what environment the current website is in and loads the relevant config file for that environment.

By default the environment is defined by the hostname, though you can also set this as an environment variable.

Config files are then loaded according to the current environment. There is support for loading a local config file for sensitive data, which is intended to not be committed to version control.

Config files

Up to three different config files are loaded:

1. **Default configuration** (in `wp-config.default.php`, e.g. shared settings such as `$table_prefix`)
2. **Environment configuration** (in `wp-config.{ENVIRONMENT}.php`, e.g. any setting specific to the environment such as database name or debug mode)

-
3. **Optional local settings** (in `wp-config.local.php`, e.g. any sensitive settings you do not want to commit to version control, e.g. database password)

Environment values

By default, environment values are:

- `production` (live website)
- `staging` (test website for client review)
- `development` (local development copy of the website)

You can add other environment values by adding these to the `wp-config.env.php` file.

Setting the environment

The current environment is detected in one of three ways:

Environment variable

You can set an environment variable called `WP_ENV` to set which environment the website uses in your webserver configuration.

This is commonly done via Apache in your virtual host declaration:

```
1 SetEnv WP_ENV production
```

If you don't use Apache consult your webserver documentation.

Server hostname

The current environment can also be detected from matching the hostname with the domain setup in `wp-config.env.php`.

WP-CLI

If you're using WP-CLI you can specify your environment using an `.env` file.

You need to create a file called `.env` and simply write the current environment in this file as a string.

Example:

1 development

This file needs to be placed among the `wp-config*.php` files (this applies if you keep these files in a sub-folder and the `wp-config.php` file in the web root).

It is recommended you do not add this file to version control.

The `wp-config.env.php` file

You need to edit the `wp-config.env.php` file to define some settings related to the current environment URL. This needs to be set regardless of which method is used to set the environment, since all methods set the WordPress URL via settings contained in this file.

This file contains a simple array, made up of:

```
1 environment names =>
2     domain  => The domain name.
3             This can also be an array of multiple domains.
4             You can also use a wildcard * to indicate all sub-
5             domains at a domain, which is useful when using
6             WordPress Multisite. If you use wildcards, set the
7             domain should to a single string, not an array.
8     path    => If WordPress is installed to a sub-folder set it here.
9     ssl     => Whether SSL should be used on this domain. If set, this
10              also sets FORCE_SSL_ADMIN to true.
```

Example usage:

```
1 $env = [
2     'production' => [
3         'domain' => 'domain.com',
4         'path'   => '',
5         'ssl'    => false,
6     ],
7     'staging'   => [
8         'domain' => 'staging.domain.com',
9         'path'   => '',
10        'ssl'    => false,
11    ],
12    'development' => [
13        'domain' => 'domain.local',
14        'path'   => '',
15        'ssl'    => false,
16    ],
17 ];
```

If you use localhost for your local test website, just set the development hostname case to `localhost` rather than `domain.local`.

Example usage when setting a sub-folder, and also serving the live site via SSL:

```
1     'production' => [  
2         'domain' => 'domain.com',  
3         'path'   => 'blog',  
4         'ssl'    => true,  
5     ],
```

Example usage for using more than one domain for an environment.

```
1     'production' => [  
2         'domain' => ['domain.com', 'domain2.com'],  
3         'path'   => '',  
4         'ssl'    => false,  
5     ],
```

Example usage when using a wildcard for WordPress multi-site.

```
1     'production' => [  
2         'domain' => '*.domain.com',  
3         'path'   => '',  
4         'ssl'    => false,  
5     ],
```

Installing

Please note this requires PHP5.4 or above. You should really be on PHP5.6 at a minimum!

1. Download the required files via `wordpress-multi-env-config.zip`
2. First make a backup of your existing `wp-config.php` file.
3. Copy the following files from this repository to your WordPress installation:

```
1 wp-config.default.php  
2 wp-config.env.php  
3 wp-config.php  
4 wp-config.load.php
```

3. Set the correct environments you wish to support via the file `wp-config.env.php`, see the documentation above.
4. Create one `wp-config.{environment}.php` file for each environment. You can use the sample files provided in this repository:

```
1 wp-config.development.php  
2 wp-config.production.php  
3 wp-config.staging.php  
4 wp-config.local.php
```

-
5. Review your backup `wp-config.php` file and copy config settings to either the default config file or the environment config files as appropriate. It is suggested to:
 - If the setting is the same across all environments, add to `wp-config.default.php`
 - If the setting is unique to one environment, add to `wp-config.{environment}.php`
 - If the setting is sensitive (e.g. database password) add to `wp-config.local.php`
 6. Remember to update the authentication unique keys and salts in `wp-config.default.php`
 7. If you use version control exclude `wp-config.local.php`, an example below for Git:

```
1 # .gitignore
2 wp-config.local.php
```

You should now be able to load up the website in each different environment and everything should work just fine! It should now be safe to delete your backup `wp-config.php` file.

Moving your config files outside of the document root

If you want to store your config files outside of the document root for additional security, this is very easy.

Simply move all the config files except for `wp-config.php` itself into another folder (which can be outside the doc root). Next amend the require path for `wp-config.load.php` in `wp-config.php` to point to the new location and everything will work just fine!

Example directory structure:

```
1 config/
2     wp-config.default.php    (Config folder outside of doc root)
3     wp-config.development.php
4     wp-config.env.php
5     wp-config.load.php
6     wp-config.local.php
7     wp-config.production.php
8     wp-config.staging.php
9 web/
10    wp-config.php            (Your website doc root, where WordPress
                             is installed)
```

Example `wp-config.php`

```
1 /** Load the Studio 24 WordPress Multi-Environment Config. */
2 require_once(ABSPATH . '../config/wp-config.load.php');
```