Develop with Git: grow from user to developer

The recommended way to build websites with Exponential. Install the kernel and every se7enxweb extension as Git clones with their full history, turn an existing site into clones, keep them up to date, and send your…

Source file: `doc/eng-us/help/develop-with-git.md` Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/develop-with-git.md

You know your way around the administration, you have published content, maybe you have changed a template. This page helps you take the next step: working on the code of your site the way the people who build Exponential do. The tool for that is **Git**, and the most useful single decision you can make is to install Exponential **as Git clones with their full history**, the kernel and every extension, instead of as release archives.

> **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.

Nothing here needs you to be an expert. Each step shows the command to type and what you should see, and every command on this page was tried on a fresh machine before it was written down.

Why Git clones and not archives

Composer can install a package in two ways:

	Release archive ("dist")	Git clone ("source")
What you get	the files of one version, nothing else	the files **and** the complete history of the project
Size and speed	small, fast	larger, a few minutes more on the first install
See what you changed	`composer status -v` names changed files	`git status` and `git diff` show every change, line by line

	Release archive ("dist")	Git clone ("source")
Find out why a line is there	not possible	`git log` and `git blame` show who changed it, when and why
Find which change broke something	not possible	`git bisect` finds it in a few steps
Use a fix before it is released	copy files by hand	`git cherry-pick` or check out the commit
Keep your own changes	lost at the next update	kept on your own branch; Composer refuses to overwrite them
Send a fix back to the project	not possible	push a branch to your fork and open a pull request

Archives are fine for a server that only runs a finished site. For **building** a website, Git clones are the recommended way: they turn every package into something you can read, search, understand and improve.

What you need

- **Git** 2.28 or later (`git --version`).
- **Composer** 2 (`composer --version`), and PHP as listed in Requirements.
- A free **GitHub account**, only when you want to send changes back. Installing needs none.

The kernel is github.com/se7enxweb/exponential, and every extension has its own repository in the se7enxweb organization. They are published on Packagist under the same names, so Composer can fetch each one either as an archive or as a clone.

Step 1: tell Composer to prefer clones

One setting makes Composer install every `se7enxweb/*` package as a Git clone, and everything else (the Zeta Components, the libraries) as a fast archive. Set it once for your user account:

Shell

```
composer config --global 'preferred-install.se7enxweb/*' source
```

Keep the quotes around `se7enxweb/*`, so that your shell does not try to expand the `*`. To see the setting:

Shell

```
composer config --global preferred-install
```

```
{"se7enxweb/*":"source","*":"dist"}
```

If you prefer to keep the setting with one project instead (for example so that everyone on your team gets it), run the same command without `--global` inside the project. It is then written into that project's `composer.json`.

Step 2: install with Git

A new site

Shell

```
git clone https://github.com/se7enxweb/exponential.git exponential
cd exponential
git switch -c my-site
composer install
```

- `git clone` fetches the kernel with its whole history: more than 27,000 commits, a download of about 370 MB. On a normal connection that takes two to three minutes.
- `git switch -c my-site` starts **your own branch** for your site's changes, on top of `main`, the current development line. To build on a release instead, name its tag: `git switch -c my-site v6.0.14`. The release tags are listed with `git tag -l 'v*' --sort=version:refname`; read Versions, tags and branches before choosing one, because older tags need a newer PHP for their test tools.
- `composer install` then clones every extension the kernel requires into `extension/<name>`, each with its own history (on `main` that is 67 repositories, a few minutes), and fetches the libraries into `vendor/` as archives.

Then install the site as usual: Choosing an install method or the one command `php bin/php/console exp:install --apply` from the console install.

For a local development site under Apache, put the rules and the constants into place from the console. A copy of `config.php-RECOMMENDED` sets nothing, so the built-in defaults apply, which suit development (INI changes are seen at once); `exp:config:list` shows what could be set:

Shell

```
php bin/php/console exp:setup:status
php bin/php/console exp:setup:htaccess --apply
php bin/php/console exp:setup:config --apply
php bin/php/console exp:config:list
```

The clone tracks only the templates (`.htaccess_root`, `.htaccess_root_static`, `config.php-RECOMMENDED`); the copies show up as untracked files in `git status`. Never commit `config.php`: it can hold a password. In a repository of your own site, add `/config.php` to its `.gitignore`. Details: Setting up .htaccess and config.php from the console.

Prefer `composer create-project`? It can make clones too, but you need two options together:

Shell

```
composer create-project se7enxweb/exponential exponential --prefer-source --keep-vcs
```

Without `--keep-vcs`, Composer deletes the kernel's `.git` folder at the end and you are left without history. Without a version, Composer picks the newest release, so the kernel is then a clone sitting on that release's tag ("HEAD detached at v6.0.14"); run `git switch -c my-site` in it before you change anything. Here `--prefer-source` makes every package a clone, the libraries included; with the setting of step 1 you can leave it out and get clones for the `se7enxweb/*` packages only.

Check that it worked

Shell

```
composer status -v
ls -d extension/*/.git | wc -l
git -C extension/eztags log --oneline -3
git -C extension/eztags status
```

`composer status -v` should say `No local changes`, the second command counts the extensions that are clones, and the last two show the newest commits of one extension and where it stands:

```
HEAD detached at v2.4.21
nothing to commit, working tree clean
```

"HEAD detached" is normal. Composer checks out the exact release tag your `composer.json` asks for, not a branch. You can read, search and test as much as you like in that state. When you want to change something, start a branch first (see Contribute a change with a pull request).

`git status` in the installation root lists the extensions as untracked folders: they are separate repositories, not part of the kernel's, and that is expected.

Step 3: turn an existing site into clones

A site that was installed from archives can be moved over without reinstalling it. **Make a backup first** (with git_manager's backup manager, or a copy of the folder and the database), and try it on a copy of the site before the live one.

The kernel

Find out which version you run, then connect the folder to the kernel's repository at exactly that version:

Shell

```
php bin/php/console --version
git init -b my-site
git remote add origin https://github.com/se7enxweb/exponential.git
git fetch origin --tags
git reset v6.0.14
git status
```

Use your own version in place of `v6.0.14`; a site installed from `main` uses `git reset origin/main`. `git reset` without `--hard` only moves Git's pointer: **it does not change a single file**. `git status` then shows exactly what differs from the release, which on a site installed from an archive means your own changes and the files your installation created (such as `settings/override/`; most of `var/` is ignored):

```
On branch my-site
 M design/standard/templates/pagelayout.tpl
?? settings/override/
```

Commit the changes you want to keep on `my-site`. The files in `settings/override/` and `settings/siteaccess/` hold database passwords: never push them to a public repository.

The extensions

Shell

```
composer config --global 'preferred-install.se7enxweb/*' source
composer status -v
composer reinstall 'se7enxweb/*'
composer status -v
```

`composer reinstall` removes each `se7enxweb/*` package and installs it again as a clone, at the same version.

> **Warning**
>
> `composer reinstall` deletes the package's folder first. If the first `composer status -v` lists changed files in an extension that is still an archive, copy that folder somewhere safe before you reinstall, then compare it with the clone (`diff -ru <copy> extension/<name>`) and bring your changes back on a branch.

After the reinstall, regenerate the class maps and clear the caches as after any code change:

Shell

```
php bin/php/ezpgenerateautoloads.php -e
php bin/php/ezcache.php --clear-all
```

The administration can do the same without a shell: the ezupdate extension (**Setup** > **Updates and packages**) lets you choose "dist or source" for each install or update run.

Contribute a change with a pull request

Found a bug and fixed it? Send the fix back, so that it is in the next release and you do not have to carry it yourself. This works the same for the kernel and for any extension; the example uses `eztags`.

1. On GitHub, open the repository (github.com/se7enxweb/eztags) and press **Fork**. You now have your own copy, `github.com/<you>/eztags`.
2. Add your fork as a second remote and start a branch:

   Shell

   ```
   cd extension/eztags
   git remote add fork https://github.com/<you>/eztags.git
   git switch -c fix-tag-search
   ```

3. Make the change, test it on your site, and commit it. Start the message with what the change does: `Fixed:`, `Added:`, `Updated:`, `Removed:` or `Renamed:`.

   Shell

   ```
   git commit -am "Fixed: The tag search finds tags with an apostrophe"
   git push -u fork fix-tag-search
   ```

4. GitHub prints a link to open a **pull request**; follow it, describe what you fixed and how you tested it, and send it. (With the GitHub command line tool: `gh pr create`.)

A clone made by Composer already has a remote named `origin` that points at the project; pushes go to your fork instead, so you need no write access to the project. For the kernel itself,

read `CONTRIBUTING.md` in the root of the installation first: it explains the commit hook (`make hooks`) and the file manifest (`make manifest-fix`) that every kernel pull request keeps current.

Expect many review rounds. 7x reviews a commit or pull request in many more rounds than usual, sometimes hundreds of revisions; be ready to revise your change until it is simple to accept. A fix for behavior that turns out to be deliberate, a negative feature other sites depend on, needs a clear justification on top, and an option is easier to accept than a new default. While the pull request is open, squash it and rebase it onto `main` as the reviews ask, which 7x requires:

Shell

```
git fetch origin
git rebase -i origin/main
git push --force-with-lease
```

Push with force only to your own pull request branch. After the merge, `main` is never rewritten (Code reviews).

Copyright assignment (CLA). The Exponential Foundation and 7x merge a code contribution to the core kernel (`se7enxweb/exponential`), sent in any form (pull request, patch, issue or mail), only after you have signed the Contributor Licensing Agreement. **By signing and submitting, you assign the copyright in your contribution to the Exponential Foundation and 7x, which become its owners, and give up control over it:** The Exponential Foundation and 7x may use, change, relicense and distribute it in any way, also under other licenses and in commercial products, without asking or paying you, and you cannot withdraw it or set conditions on it. Your own custom extensions, and the branch of your own site, are not affected. See Copyright assignment (CLA) and the CLA page.

Built an extension of your own? Contribute extensions back shows how to publish it on GitHub and Packagist so every site can install it, and what the GPL requires when you hand an extension to others.

Improving the documentation works the same way: these help pages are the files in `doc/` of the kernel. See Free documentation for what you may do with them.

Everyday work with clones

See what changed

Shell

```
git status
git diff
composer status -v
```

`composer status -v` looks into every package at once and names the ones with uncommitted changes, unpushed commits or a different version than the lock file says.

Find out why and when

Shell

```
git log --oneline -10 -- kernel/content/view.php
git blame kernel/content/view.php
git log -S "CheckValidity" --oneline
```

The first shows the last ten changes to one file, the second who last changed each line, the third every commit that added or removed a piece of text. In an extension, run the same commands inside `extension/<name>` or with `git -C extension/<name> ...`.

Find the change that broke something

Shell

```
git bisect start
git bisect bad
git bisect good v6.0.13
```

Git checks out a version halfway between; test it and answer `git bisect good` or `git bisect bad` until Git names the commit. `git bisect reset` brings you back to where you started.

Use a fix before it is released

When a fix is already in the project's history but not yet in a release:

Shell

```
git -C extension/eztags fetch origin
git -C extension/eztags switch -c with-fix
git -C extension/eztags cherry-pick <commit>
```

While that branch exists only on your machine, Composer refuses to update the package (see When Composer refuses): push the branch to your fork, or go back to the release with `git -C extension/eztags switch --detach v2.4.21` (your release tag) once a release contains the fix.

Keep up to date

The kernel, on your branch, then the extensions to match:

Shell

```
git fetch origin --tags
git merge origin/main
composer update
```

(or merge the newest release tag instead of `origin/main`). Merging keeps your own commits on `my-site`; `composer update`, not `composer install`, because the kernel's `composer.json` changed and your lock file has to follow it. Always Git first (the kernel), then Composer (the extensions). Update with Git has the whole routine, on the command line (Track A) or in the administration with Git Manager and Updates and packages (Track B), for the schedule you choose (hourly, daily, weekly, monthly or only at each release), how to see what changed, and an automatic update. Moving the code is half of an upgrade; Upgrading has the database half.

One extension on its own, after reading its release notes:

Shell

```
composer update se7enxweb/eztags
```

Composer moves the clone to the new tag. Its history stays, and so do your branches.

When Composer refuses

Composer protects your work in a clone. These are the messages you will meet, and what to do:

Composer says	Why	What to do
`Source directory ./extension/eztags has uncommitted changes.`	you changed files and did not commit them	commit them on a branch, or `git -C extension/eztags stash`, run the update, then `git -C extension/eztags stash pop`
`Source directory ./extension/eztags has unpushed changes on the current branch`	you committed on a branch that exists only on your machine	`git push -u fork <branch>` to your fork, then update; your branch is kept
`You have version variations in the following dependencies (from composer status)`	the clone is on a branch, not on the tag in the lock file	expected while you work on a fix; `composer update <package>` puts it back on the tag
the package is still installed from an archive	it was installed before the setting of step 1: Composer does not convert packages that are already installed	`composer reinstall <package>` (see step 3)

Three more things worth knowing:

- **Never `git clone` by hand into a folder Composer installed as an archive.** Composer still thinks the folder is an archive and, at the next update of that package, replaces it without asking: your clone, its history and your commits are gone. Let Composer make the clone with `composer reinstall <package>` (see step 3).
- An update can stop half-way when it refuses: the lock file may already name the new version while the folder still has the old one. Fix the cause, then run `composer install` to finish.
- `composer update` moves **every** package to the newest version its constraint allows. On a live site, update one package at a time (`composer update se7enxweb/<package>`), as Getting the code explains.

Where to go next

- Update with Git: stay in sync with GitHub on the schedule you choose.
- Getting the code: every way to install, versions and tags, extensions as Composer packages.
- Extensions and the extension pages.
- Setting up .htaccess and config.php from the console: the local `.htaccess` and `config.php`.
- git_manager: see and change the state of a Git checkout from the administration, and make backups before you change anything.
- When something goes wrong: which issue tracker to use and how to write a good report. The **Report an error** page of this help prepares the issue for you.
- The project on GitHub: se7enxweb/exponential, its issues, and every se7enxweb repository.
- Free documentation: these pages are free to copy, translate and improve.

- Git itself: the free book Pro Git, and GitHub's guide to pull requests.

See also: Ecosystem, PHP Developers, AI, agents and agentic, Software Engineers, Web Framework Developers, CMS Developers, DXP Developers, Integrators.

> **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.