interweb

Generated on 15/11/2026 00:10

KNOWLEDGE BASE · PUBLIC DOCUMENT

Git in cPanel: publishing your site from a repository

Frameworks, Composer & Git

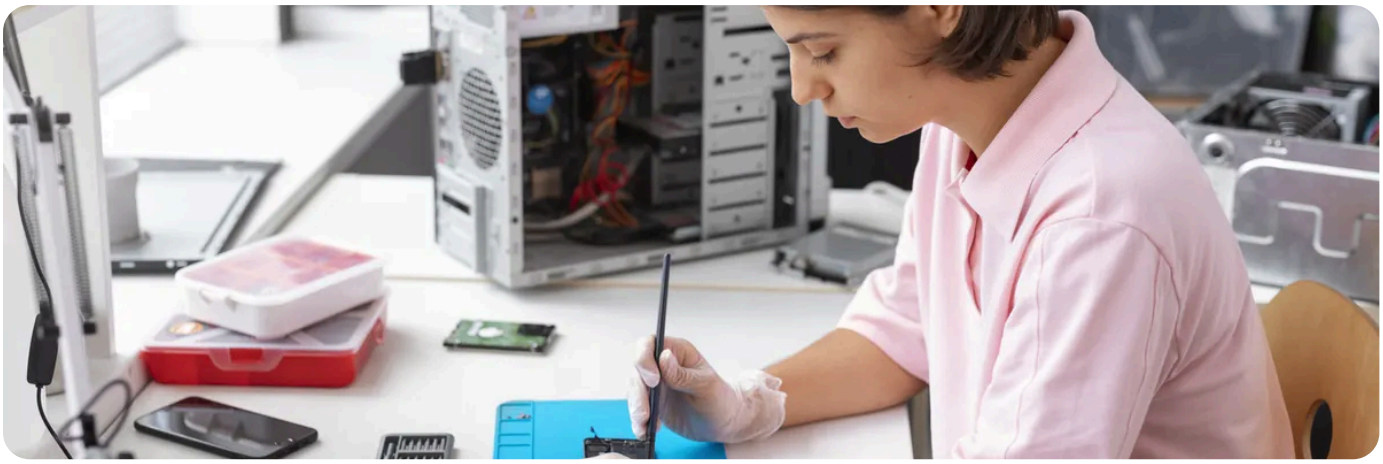
CODE	VERSION	COUNTRY · LANGUAGE
RLX-AO-KB-678	v2026.10.18	Angola · English

Interweb Angola
RLX Angola, Lda. · Group RLX Holdings
Tax ID (NIF) 500 260 5724 · Commercial registration no. 26464-25/250724

https://www.meu.interweb.ao/knowledgebase/678

Git in cPanel: publishing your site from a repository

Frameworks, Composer & Git · Created on **September 27, 2026** · Updated on **October 18, 2026** · 4 min read



Anyone who works with **Git** knows the nuisance: the code is in a repository, the site is on the server, and between the two there is always somebody dragging files over FTP and forgetting one. cPanel has a tool that closes that gap.

It is called **Git Version Control** and it sits under **Files** in the panel. The Git installed on this server is version `2.48.2`.

> **This is not a code hosting service.**
> It does not replace GitHub, GitLab, Bitbucket or your company's own Git server. There is no page to browse code in, no merge requests, no team management. The central repository still belongs to somebody else; what this tool does is **pull from there to here** and **publish**.

What the tool does, in two parts

Part	What happens
The repository	It creates a repository inside your account. It can be empty, to start from nothing, or **cloned** from an address you give it. Once it exists, you can ask it to fetch new changes from the remote repository.
The deployment	It copies files from the repository to wherever the site reads them, following the instructions in a file called `.cpanel.yml` that you put at the root of the repository. **Without that file the repository exists but deploys nothing.**

That split matters: the repository can live in a folder outside the website, and only what the deployment copies ends up visible on the internet. That is exactly how you want it.

The order that works

1. **Decide where the repository lives, and keep it out of the public website folder.** A folder beside it, in the account home, is right. The reason is in the warning below, and it is worth reading before you choose.

2 **Clone the remote repository** by giving it the address. If the repository is private, the server needs credentials to reach it, and that does not sort itself out: either the repository is public, or you arrange access keys on the side that hosts it.

3 **Write the `.cpanel.yml` file** at the root of the repository, with the lines that say what to copy and where to. It is a short text file, it goes into the repository like any other, and cPanel's own documentation shows the exact format.

4 **Pull the changes** from the remote repository into the one in your account whenever there is something new.

5 **Run the deployment**, then check the site. The tool keeps the result of the last deployment, which is where you see whether it finished or where it stopped.

! **Never leave the `.git` folder inside what the site serves.**
If the repository sits in the public folder, anybody can download your entire code history from a browser, and with it the passwords somebody committed to a config file two years ago and later removed. Removed from the file, still in the history. If yours is already like that, move it out today and change every password that appears in the history.

What should never go into the repository

Keep out	Why
The config file with the database connection	It carries a username and a password. It stays in the history for ever and travels with the repository to everyone who clones it. Keep it out of the repository and leave it on the server.
The folder of files visitors upload	Images, attachments, documents. They are not code, they grow on their own, and a deployment that overwrites them wipes everything users have sent since last time.
The database	Git handles files. The database is copied separately: **making and keeping your own backup**.
Downloaded dependency folders	They are enormous and they rebuild themselves from the file that lists them. Committing them makes every operation slow for no gain at all.

Try it on a subdomain first.
A badly written deployment copies over real files. Make a staging subdomain (**how to add another domain or a subdomain**), point the first deployment there, and only then switch to the live site.

When this is worth it, and when it is not

Case	Worth it?
A hand-built site, or your own application, with more than one person touching it	**Very much so.** It ends the «who overwrote this» conversation and the files forgotten in transit.
A normal WordPress, where the content is written in the dashboard	**Almost never.** What changes in a WordPress is the database and the uploaded files, and neither belongs in Git. There the answer is backups and a staging site.
A theme or plugin you wrote yourself	**Yes.** That really is code, and it can live in a repository of its own.
Publishing a static site built on your computer	**Yes.** This is one of the cleanest cases: the repository brings the finished folder and the deployment copies it across.

If you would rather keep sending files by hand, none of this is compulsory: **FTP accounts and connecting with FileZilla** and **day to day in FileZilla** still do the job.

For the command-line work that goes with a repository, the panel also has **Terminal**, where the account allows it.

NEED A HAND?

Want to deploy from a repository and not sure where to start? Tell us what you have.

Open a support ticket →

SEE ALSO

- **Hosting plans and what each one includes** →
- **Support Policy: how far our help goes** →

interweb

Interweb Angola

RLX Angola, Lda. · Group RLX Holdings

TAX ID (NIF)	500 260 5724
COMMERCIAL REGISTRATION NO.	26464-25/250724
REGISTERED OFFICE	Rua Direita do Patriota, Bairro Dangereux Belas, Luanda, Angola
CONTACT	info@interweb.ao · +244 942 011 191 · www.interweb.ao
OPENING HOURS	Monday to Friday - 08 AM to 04 PM Saturday & Sunday - Closed

Online version of this document

RLX-AO-KB-678 · v2026.10.18 · Generated on 15/11/2026 00:10