

Managing File and Configuration Revisions with Git



- [Goal](#)
 - [Prerequisite knowledge](#)
 - [Site prerequisites](#)
 - [Steps](#)
 - [Initializing the repository](#)
 - [Updating files in the repository](#)
 - [Making a copy of the files in your repository](#)
 - [Managing configuration in the repository](#)
 - [Related concepts](#)
-

Goal

Use the Git revision control tool to manage revisions to files and configuration on your site.

Prerequisite knowledge

- [Concept: Additional Tools](#)
- [Concept: Methods for Downloading and Installing the Core Software](#)
- [Concept: Development Sites](#)

- How to set up a Git repository and find its *clone* URL. For example, if you want to use GitLab to host your repository, see [GitLab "Create a project" page](#) and [GitLab "Command Line basic commands" page](#). And if you want to use GitHub to host your repository, see [GitHub "Create a repo" page](#) and [GitHub "Which remote URL should I use" page](#).
- How to open and use a command terminal window and a plain-text editor.
- To manage configuration, how to unpack and pack archive files (such as *.zip* and *.tar.gz*).

Site prerequisites

- You must have downloaded the software for your site, using one of the methods in [Concept: Methods for Downloading and Installing the Core Software](#). If you want to manage configuration, you must have installed the software and have a running site.
- Git client software must be installed on your site's server. See [Git](#) for instructions.
- You must have a new repository created and know its Git clone URL.

Steps

Initializing the repository

Do these steps once, after creating a Git repository, to connect your local directory to the repository and add the initial files to it.

1. Open a command terminal window, and change to the directory where your site's files are located. This is your "top" directory.
2. Determine where your web root is. If the *core*, *modules*, and *themes* directories are located directly in this directory, then you are in your web root. If you have used Composer to download the software, then these files are located inside the *web* subdirectory (which is your web root).
3. In a plain text editor, create a new file called *.gitignore* in the top directory (or edit it if it already exists). This file contains a list of files and directories that Git should ignore (not add to the repository). For example, the *settings.php* file for your site should not be added to Git, because it contains your database account information, and the media files uploaded to your site (usually in *sites/default/files*) should not be in Git either -- the objective is to have the software in the repository, not the site content.
4. Make sure the following lines are in the *.gitignore* file. If your web root is not your top directory, check each of these to see if they need a prefix. For example, *sites* may need to be replaced with *web/sites*.

```
sites/*/settings*.php
sites/*/files
config
```

5. Enter the following commands:

```
git init
git add -A
```

6. Optionally, verify the list of files you will be adding to your Git repository by entering this command and scrolling through the (very long) list:

```
git status
```

7. Enter the following commands. You can substitute your own *commit message* for "Initial file add" if you wish, and you will need to substitute your own Git clone URL for the URL in the second command:

```
git commit -m "Initial file add"
git remote add origin https://gitlab.com/example-name/example-repo.git
git push -u origin main
```

8. If you are using GitLab, GitHub, or another host with online access, you can now go to your repository page and see that the files are there.

Updating files in the repository

Use these steps when you have updated, added, or deleted one or more files in your site, in order to send the changes (*push*) to your repository.

1. Open a command terminal window, and change to the directory where your site's files are located.
2. Check the list of files that have been added, changed, or deleted:

```
git status
```

3. Optionally, for text files that have been changed (not images), look at the differences between the new and old versions of the file:

```
git diff path/to/file.txt
```

4. Stage all the changes for the next commit, and verify that they are staged:

```
git add -A
git status
```

5. You can omit a particular file from the commit that you have already staged, or add another file to the ones you have already staged. If a particular file or directory keeps getting added by mistake, consider adding it to the `.gitignore` file so that it will be ignored by Git. Omit/add commands:

```
git reset HEAD path/to/file.txt  
git add path/to/file.txt
```

6. Commit and push your changes. Substitute something meaningful for the commit message:

```
git commit -m "commit message here"  
git push
```

7. If you have other copies of your repository, update them by opening a command window in the directory of each copy and running the following command:

```
git pull
```

Making a copy of the files in your repository

Follow these steps if you want to copy all the files in your repository to a new location. For example, you might have both a local development copy of your site and a production site, or several team members might all have local copies of the site.

1. Open a command window in the location where you want the files to be.
2. Enter the following command, substituting your repository clone URL for the URL, and the name of the subdirectory you want them in for *dirname*:

```
git clone https://gitlab.com/example-name/example-repo.git dirname
```

Managing configuration in the repository

1. Follow the instructions on [Synchronizing Configuration Versions](#) to export a complete archive of your site's configuration.
2. If you have not already initialized configuration in the repository, unpack the configuration archive into a new directory, preferably above the web root directory, and follow the instructions above to add these files to your repository.
3. After initializing, whenever your site configuration changes, export and unpack the configuration archive in the same location. Follow the instructions above to update these

files in your repository.

4. To import updated configuration to another site, make an archive of the configuration directory from your repository. Then follow the instructions on [Synchronizing Configuration Versions](#) to upload and import this archive into the site.

Related concepts

[Concept: Development Sites](#)

☐☐ #3

☐ cocobook-sync ☐☐ 2026-10-01 00:16:21 UTC

☐ cocobook-sync ☐☐ 2026-10-09 00:27:48 UTC