Using .gitignore

*Last Updated: 05 November 2014*

[**git**](https://devcenter.heroku.com/tags/git) [**gitignore**](https://devcenter.heroku.com/tags/gitignore) [**ignore**](https://devcenter.heroku.com/tags/ignore)

Table of Contents

* [Ignoring directories](https://devcenter.heroku.com/articles/gitignore#ignoring-directories)
* [Ignoring SQLite files](https://devcenter.heroku.com/articles/gitignore#ignoring-sqlite-files)
* [Further reading](https://devcenter.heroku.com/articles/gitignore#further-reading)

The source code for your application, its dependencies, small static assets (CSS, images), and most config files should be checked into your Git repo for deploy to Heroku.

Anything else, such as log files, large static assets, or SQLite database files should be ignored via one or more .gitignore files in your repo. This will keep the [slug size](https://devcenter.heroku.com/articles/slug-compiler) down, speeding up the speed at which new dynos can be started.

[Ignoring directories](https://devcenter.heroku.com/articles/gitignore#ignoring-directories)

When you [deploy with Git](https://devcenter.heroku.com/articles/git), a branch of your repository is pushed to Heroku. To ensure that superfluous assets aren’t sent, such as a logor tmp directory, configure your Git to ignore those particular assets, and remove them from your repository. The configuration takes place in a .gitignore file.

Some language frameworks automatically generate a .gitignore file - ensuring that any files that match the patterns in the file are not considered for addition to a repository. You may already have a.gitignore in the root of your application folder, which matches certain patterns - yet still want to configure it to ignore additional folders.

Let’s assume you need to ignore the contents of the log and tmpdirectories. In this example, we’ll use the approach of ignoring \*.logwithin the log folder, and ignoring the tmp folder altogether:

$ git rm -r -f log

rm 'log/development.log'

rm 'log/production.log'

rm 'log/server.log'

rm 'log/test.log'

$ git rm -r -f tmp

fatal: pathspec 'tmp' did not match any files

$ mkdir log

$ echo '\*.log' > log/.gitignore

$ git add log

$ echo tmp >> .gitignore

$ git add .gitignore

$ git commit -m "ignored log files and tmp dir"

[Ignoring SQLite files](https://devcenter.heroku.com/articles/gitignore#ignoring-sqlite-files)

If you use SQLite for your local database (which is the default for some language frameworks), you should ignore the resulting database files in the same way. Here’s one way to do so:

$ git rm -f db/\*.sqlite3

$ echo '\*.sqlite3' >> .gitignore

$ git add .gitignore

$ git commit -m "ignored sqlite databases"

[Further reading](https://devcenter.heroku.com/articles/gitignore#further-reading)

* Git Community Book on [.gitignore](http://git-scm.com/docs/gitignore)