# IGNORING IN GIT: HOW TO USE .GITIGNORE FILES


When making commits to any Git repository, you'll choose the files you want to stage and then you'll commit them.

But you might not want to commit every single one of your files—there are files that never need to get committed. This is where the .gitignore file is useful: it tells Git exactly which files to ignore and never track.

Let's take a look.

## What is gitignore?

When you're working in your copy, Git watches every file in and considers it in three ways:

1. Tracked: You've already staged or committed the file.
2. Untracked: You've not staged or committed.
3. Ignored: You've explicitly told Git to ignore the file(s).

The .gitignore file tells Git which files to ignore when committing your project to the GitHub repository. gitignore is located in the root directory of your repo.

The .gitignore file itself is a plain text document. Here's an example .gitignore file:

```
# Binaries for programs and plugins
*.exe
*.exe~
*.dll
```

```
*.so
*.dylib

# Test binary, built with `go test -c`
*.test

# Output of the go coverage tool, specifically when used with LiteIDE
*.out

# Dependency directories (remove the comment below to include it)
vendor/
```

- * is used as a wildcard match *.exe will ignore any file with the .exe extension
- / will ignore directories with the name. vendor/ ignores the vendor directory.
- # will comment the line
- will ignore values with any of the values.
    - *. ignores files file.a, file.b, file.c.
    - *.d] the dash will include a range, in this case, file extensions a-d.

# Why do I need to ignore files in Git?

You may want to ignore certain files for multiple reasons:

- The files contain sensitive data.
- The files are system specific and do not need to exist on every machine's copy.
- Excluding the files maintains system security rules and privileges. (Remember, Git repos only contain the files necessary to get tech support—not to share the entire software.)

Julien Danjou points out there is a global .ignore file for your computer to ignore for every commit:

*Not everybody uses your editor or favorite pet tool, and nobody cares. The repository you're working in is shared with a lot of other developers. Sending pull requests to just add this kind of entry to ignore files generated by your pet editor is wrong and annoying.*

# Items to put in the .gitignore

Not everything goes into gitignore, but here are common items to ignore:

## System-specific files

System-specific files need to get ignored. But, you can add these files to a global ignore file instead of the repo's ignore file.

## Vscode workspaces

Items like a vscode workspace need to be ignored.

## Security and API keys/secrets

For security, the security key files and API keys should get added to the gitignore. (That is, if they're even stored in the directory). Every commit is recorded in the history of a GitHub repo. If a key is submitted, even if it is taken down immediately after, a record of the key exists in that commit.

# How to create the global gitignore for your system

The global ignore file can go anywhere. Usually it is kept in the User's home directory.

1. Create the file:

   ```
   touch ~/.gitignore_global
   ```

2. Add the file to the Git configuration:

   ```
   git config --global core.excludesfile ~/.gitignore_global
   ```

3. Edit the file with your text editor and add your rules to it.

# See all ignored files

```
git status –ignored
```

# Getting started with gitignore

Different coding frameworks generate their own extra files. .gitignore templates can be used as a baseline to get rid of the general files most people choose to ignore.

- Python has its own files.
- System administrators have their own files.
- MacOS and Windows each have their own type of ignore files.
- Different IDEs, like VScode and Atom, have their own .gitignore files.

To get started with one that might suit your needs:

1. Explore templates. Github provides a repo of .gitignore templates
2. Search project files. Toptal allows you to search for .gitignore files based on your project
3. In the coding fashion, Googling ".gitignore for " can get you there, too.

# Additional resources

For more on related topics, explore the BMC DevOps Guide.