About large files on GitHub - GitHub Docs

5-6 minutes

GitHub limits the size of files you can track in regular Git repositories. Learn how to track or remove files that are beyond the limit.

About size limits on GitHub

GitHub tries to provide abundant storage for all Git repositories, although there are hard limits for file and repository sizes. To ensure performance and reliability for our users, we actively monitor signals of overall repository health. Repository health is a function of various interacting factors, including size, commit frequency, contents, and structure.

File size limits

GitHub limits the size of files allowed in repositories. If you attempt to add or update a file that is larger than 50 MiB, you will receive a warning from Git. The changes will still successfully push to your repository, but you can consider removing the commit to minimize performance impact. For more information, see "Removing files from a repository's history."

Note: If you add a file to a repository via a browser, the file can be no larger than 25 MiB. For more information, see "Adding a file to a repository."

GitHub blocks files larger than 100 MiB.

To track files beyond this limit, you must use Git Large File Storage (Git LFS). For more information, see "About Git Large File Storage."

If you need to distribute large files within your repository, you can create releases on GitHub.com instead of tracking the files. For more information, see "Distributing large binaries."

Git is not designed to handle large SQL files. To share large databases with other developers, we recommend using a file sharing service.

Repository size limits

We recommend repositories remain small, ideally less than 1 GB, and less than 5 GB is strongly recommended. Smaller repositories are faster to clone and easier to work with and maintain. If your repository excessively impacts our infrastructure, you might receive an email from GitHub Support asking you to take corrective action. We try to be flexible, especially with large projects that have many collaborators, and will work with you to find a resolution whenever possible. You can prevent your repository from impacting our infrastructure by effectively managing your repository's size and overall health. You can find advice and a tool for repository analysis in the `github/git-sizer` repository.

External dependencies can cause Git repositories to become very large. To avoid filling a repository with external dependencies, we recommend you use a package manager. Popular package managers for common languages include Bundler, Node's Package Manager, and Maven. These package managers support using Git repositories directly, so you don't need pre-packaged sources.

Git is not designed to serve as a backup tool. However, there are many solutions specifically designed for performing backups, such as Arq, Carbonite, and CrashPlan.

Removing files from a repository's history

Warning: These procedures will permanently remove files from the repository on your computer and GitHub.com. If the file is important, make a local backup copy in a directory outside of the repository.

Removing a file added in the most recent unpushed commit

If the file was added with your most recent commit, and you have not pushed to GitHub.com, you can delete the file and amend the commit:

1. Open Terminal.
2. Change the current working directory to your local repository.
3. To remove the file, enter `git rm --cached`:

```
$ git rm --cached GIANT_FILE
# Stage our giant file for removal, but leave it on disk
```

4. Commit this change using `--amend -CHEAD`:

```
$ git commit --amend -CHEAD
# Amend the previous commit with your change
# Simply making a new commit won't work, as you need
# to remove the file from the unpushed history as well
```

5. Push your commits to GitHub.com:

```
$ git push
# Push our rewritten, smaller commit
```

Removing a file that was added in an earlier commit

If you added a file in an earlier commit, you need to remove it from the repository's history. To remove files from the repository's history, you can use the BFG Repo-Cleaner or the `git filter-repo` command. For more information see "Removing sensitive data from a repository."

Distributing large binaries

If you need to distribute large files within your repository, you can create releases on GitHub.com. Releases allow you to package software, release notes, and links to binary files, for other people to use. For more information, visit "About releases."

We don't limit the total size of the binary files in the release or the bandwidth used to deliver them. However, each individual file must be smaller than 2 GiB.