Importing a Subversion repository

In this article

You can import a repository from Subversion by converting the repository to Git, then pushing the Git repository to GitHub Enterprise Server.

Prerequisites

To follow these steps, you must use a macOS or Linux system and have the following tools installed:

- Subversion
- Git, including `git-svn`
- Git Large File Storage (Git LFS) (see "Installing Git Large File Storage")

Importing a Subversion repository

1. Create a new repository on your GitHub Enterprise Server instance. To avoid errors, do not initialize the new repository with README, license, or gitignore files. You can add these files after your project has been pushed to GitHub Enterprise Server. For more information, see "Creating a new repository."
2. To confirm that Git is installed on your machine, run `git --version.`

   The output should be similar to `git version 2.40.0`.
3. To confirm that `git svn` is available on your machine, run `git svn --version`.

   The output should be similar to `git-svn version 2.40.0 (svn 1.14.2)`.

   If you can run `git` successfully but encounter an error when running `git svn`, you may need to install `git svn` separately. We recommend using Homebrew or the Ubuntu package registry, which include `git-svn` packages.
4. To confirm that Git LFS is installed on your machine, run `git lfs --version`.

   The output should be similar to `git-lfs/3.1.4 (GitHub; darwin arm64; go 1.18.1)`.
5. Check out your Subversion repository.

   For example, to check out the Logisim open source project from Sourceforge, run `svn checkout https://svn.code.sf.net/p/circuit/code/trunk`.
6. Move into the directory for your Subversion repository.
7. To get a list of authors in your Subversion project and store the list in `authors.txt`,

run the following script:

```shell
svn log -q | grep -e '^r' | awk 'BEGIN { FS = "|" } ; { print $2" = "$2 }' |
sed 's/^[ \t]*//' | sort | uniq > authors.txt
```

8. Update your `authors.txt` file, mapping the author name used in the Subversion repository to the name you want to use in your Git repository, with the following format:

   ```
   octocat = The Octocat <octocat@github.com>
   ```

9. To convert your Subversion repository to a Git repository, use `git svn`.

   - If your Subversion repository has a standard format, with “trunk”, “branches”, and “tags” folders, run `git svn clone -s URL PATH/TO/DESTINATION --authors-file PATH/TO/AUTHORS.TXT`, replacing `URL` with the URL of the Subversion repository, `PATH/TO/DESTINATION` with the path to the directory you want to clone the repository into, and `PATH/TO/AUTHORS.TXT` with the path to your `authors.txt` file.

     For example, to clone the Logisim project from Sourceforge into a directory called logisim, run `git svn clone -s https://svn.code.sf.net/p/circuit/code logisim --authors-file path/to/authors.txt`.
   - If your Subversion repository is non-standard, you can customize `git svn` to handle your repository. For more information, see git-svn in the Git documentation.

10. Git will check out each SVN revision and turn the revision into a Git commit. If your repository has many files or a lot of history, this process will take a long time.

    For large repositories, the command may freeze. If so, you can begin where you ended by terminating the command with `Ctrl`+`C`, moving to your new directory, and then running `git svn fetch`.

11. Move into the directory for the newly-created Git repository.

12. To add your GitHub repository as a remote, run `git remote add origin URL`, replacing `URL` with the URL for the GitHub repository you created earlier, such as `https://github.com/octocat/example-repository.git`.

13. To push the repository to GitHub, run `git push --mirror origin`.

    If your repository contains any files that are larger than GitHub Enterprise Server's file size limit, your push may fail. Move the large files to Git LFS by running `git lfs import`, then try again.