Syncing a fork

In this article

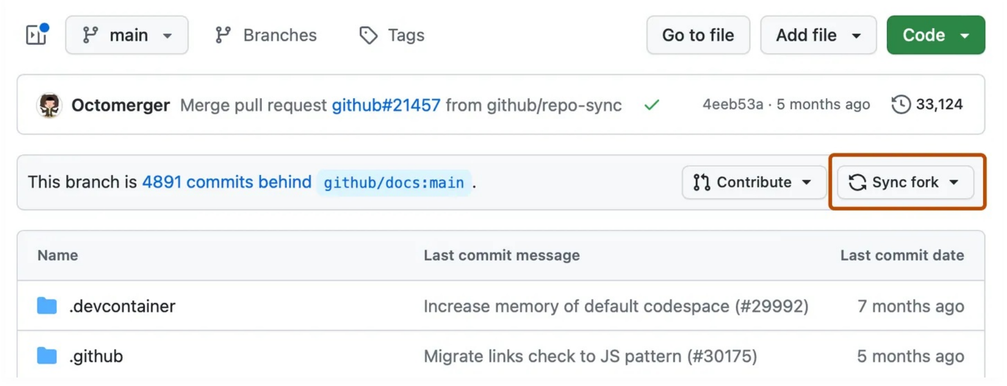
Sync a fork of a repository to keep it up-to-date with the upstream repository.

Who can use this feature
People with write access for a forked repository can sync the fork to the upstream repository.

Mac Windows Linux

Syncing a fork branch from the web UI

1. On GitHub Enterprise Cloud, navigate to the main page of the forked repository that you want to sync with the upstream repository.
2. Above the list of files, select the **Sync fork** dropdown menu.


main
Branches
Tags
Go to file
Add file
Code
Octomerger Merge pull request github#21457 from github/repo-sync ✓ 4eeb53a · 5 months ago 33,124
This branch is 4891 commits behind github/docs:main .
Contribute
Sync fork

Name	Last commit message	Last commit date
.devcontainer	Increase memory of default codespace (#29992)	7 months ago
.github	Migrate links check to JS pattern (#30175)	5 months ago

3. Review the details about the commits from the upstream repository, then click **Update branch**.

If the changes from the upstream repository cause conflicts, GitHub will prompt you to create a pull request to resolve the conflicts.

Syncing a fork branch with the GitHub CLI

GitHub CLI is an open source tool for using GitHub from your computer's command line. When you're working from the command line, you can use the GitHub CLI to save time and avoid switching context. To learn more about GitHub CLI, see "About GitHub CLI."

To update the remote fork from its parent, use the `gh repo sync -b BRANCHNAME` subcommand and supply your fork and branch name as arguments.

```
gh repo sync owner/cli-fork -b BRANCH_NAME
```

If the changes from the upstream repository cause conflict then the GitHub CLI can't sync. You can set the `--force` flag to overwrite the destination branch.

Syncing a fork branch from the command line

Before you can sync your fork with an upstream repository, you must configure a remote that points to the upstream repository in Git. For more information, see "Configuring a remote repository for a fork."

1. Open TerminalTerminalGit Bash.
2. Change the current working directory to your local project.
3. Fetch the branches and their respective commits from the upstream repository. Commits to `BRANCHNAME` will be stored in the local branch `upstream/BRANCHNAME`.

   ```
   $ git fetch upstream
   > remote: Counting objects: 75, done.
   > remote: Compressing objects: 100% (53/53), done.
   > remote: Total 62 (delta 27), reused 44 (delta 9)
   > Unpacking objects: 100% (62/62), done.
   > From https://github.com/ORIGINAL_OWNER/ORIGINAL_REPOSITORY
   >  * [new branch]      main     -> upstream/main
   ```

4. Check out your fork's local default branch - in this case, we use `main`.

   ```
   $ git checkout main
   > Switched to branch 'main'
   ```

5. Merge the changes from the upstream default branch - in this case, `upstream/main` - into your local default branch. This brings your fork's default branch into sync with the upstream repository, without losing your local changes.

   ```
   $ git merge upstream/main
   > Updating a422352..5fdff0f
   > Fast-forward
   >  README                    |    9 -------
   >  README.md                 |    7 ++++++
   >  2 files changed, 7 insertions(+), 9 deletions(-)
   >  delete mode 100644 README
   >  create mode 100644 README.md
   ```

   If your local branch didn't have any unique commits, Git will perform a fast-forward. For more information, see Basic Branching and Merging in the Git documentation.

   ```
   $ git merge upstream/main
   > Updating 34e91da..16c56ad
   > Fast-forward
   >  README.md                 |    5 +++--
   >  1 file changed, 3 insertions(+), 2 deletions(-)
   ```

   If your local branch had unique commits, you may need to resolve conflicts. For more information, see "Addressing merge conflicts."

Tip: Syncing your fork only updates your local copy of the repository. To update your fork on GitHub.com, you must push your changes.