git-filter-repo Rewrite Git history

CONCEPTS

Run it in a fresh clone: it rewrites every commit hash and refuses on repos that look used.

Fresh clone throwaway copy
↓ `filter-repo`
Rewritten history new hashes, origin removed
↓ `inspect`
Check result log, ls-files
↓ `remote add + push --force`
New remote collaborators re-clone

Filters combine: path selection, renames and content edits run in one pass over history.

ANALYZE

`git filter-repo --analyze`
Write size and path reports

`git clone --no-local <src> <dst>`
Fresh clone of a local repo

`git filter-repo --force`
Skip the fresh-clone check

`git filter-repo --dry-run`
Save before/after streams only

SELECT PATHS

`--path <path>`
Keep a file or directory

`--path <dir>/ --path <file>`
Keep several paths (union)

`--invert-paths`
Remove the matched paths instead

`--path-glob '<glob>'`
Match paths by glob

`--path-regex '<regex>'`
Match paths by regex

SELECT PATHS (cont.)

`--paths-from-file <file>`
Read path rules from a file

RESTRUCTURE

`--subdirectory-filter <dir>`
Extract a subdir as the root

`--to-subdirectory-filter <dir>`
Move the root into a subdir

`--path-rename <old>:<new>`
Rename a path or directory

CONTENT AND MESSAGES

`--replace-text <file>`
Replace strings in file contents

`--replace-message <file>`
Replace strings in messages

`--strip-blobs-bigger-than <size>`
Drop files over a size

`--strip-blobs-with-ids <file>`
Drop blobs listed by ID

`--mailmap <file>`
Rewrite authors via mailmap

`--use-mailmap`
Use the .mailmap file

AFTER THE REWRITE

`git remote add origin <url>`
Restore the removed remote

`git push --force --all`
Publish rewritten branches

`git push --force --tags`
Publish rewritten tags

`git log --oneline --stat`
Check the result

`git ls-files`
Confirm remaining files