

git-dct

Author: Adrian DC

Description: Git development CLI tools for daily usage

Date: 10/02/2025

Version: 2.1.1

► Aliases

pypi v2.1.0 python 3.9 | 3.10 | 3.11 | 3.12 downloads 275/month license Apache License 2.0

pipeline running bugs 0 code smells 0 coverage 100% lines of code 2.1k

quality gate passed

pre-commit enabled commitizen friendly gcil enabled pre-commit-crocodile enabled

Git development CLI tools for daily usage

Table of contents

- [git add](#) - Add file changes for commits
- [git branch](#) - List, create, or delete branches
- [git cherry-pick](#) - Apply changes introduced by existing commits
- [git commit](#) - Commit changes to the repository
- [git describe](#) - Describe Git objects for humans
- [git diff](#) - Show changes between commits and sources
- [git fetch](#) - Download references and objects from remotes
- [git log](#) - Show commit logs
- [git merge](#) - Merge development histories together
- [git push](#) - Push references and objects to remotes
- [git rebase](#) - Reapply and rework commits history
- [git remote](#) - Manage remote sources repositories
- [git reset](#) - Reset history or sources to a specific state
- [git show](#) - Show commit contents or other objects
- [git stash](#) - Store or stash uncommitted changes away
- [git submodule](#) - Initialize, update or inspect submodules

git add - Add file changes for commits

- **git aa** : Add files verbosely
 - **git an** : Add files as dry-run only
 - **git ap** : Add files in interactive patch mode
 - **git acae** : Add files in interactive patch mode and amend to the current commit
-

git branch - List, create, or delete branches

- **git bd** : Delete branch
 - **git br** : List, create, or delete branches
 - **git brl** : List all available branches
 - **git ch** : Switch branches or restore working tree files
 - **git sw** : Create, update and switch to a branch
-

git cherry-pick - Apply changes introduced by existing commits

- **git cp** : Apply the changes introduced by existing commits
- **git cpa** : Abort the cherry-pick operation in progress
- **git cpc** : Continue the cherry-pick operation in progress
- **git cps** : Skip the cherry-pick operation in progress
- **git fcp** : Git fetch path and cherry-pick commits

git commit - Commit changes to the repository

- **git c** : Commit changes to the repository
 - **git ca** : Amend changes to the current commit
 - **git cae** : Amend changes to the current commit without edition
 - **git cauthor** : Update commit authorship from current user or a commit
 - **git ce** : Commit empty changes to the repository
 - **git cs** : Commit changes to the repository with 'Signed-off-by'
 - **git cad** : Reset current commit date to now
 - **git sl** : Amend the last commit with interactively staged changes
-

git describe - Describe Git objects for humans

- **git tagdescribe** : Describe Git history relative to tags
-

git diff - Show changes between commits and sources

- **git diffall** : Show changes between commits and sources with all differences highlighted
- **git diffc** : Show changes between commits and sources with only char-level differences
- **git diffw** : Show changes between commits and sources with only word-level differences
- **git st** : Git history with remote comparator
- **git stat** : Git history with remote comparator
- **git di** : Show changes in sources with HEAD (or a commit)

git fetch - Download references and objects from remotes

- **git f** : Download references and objects from remotes
 - **git fe** : Git fetch with interactive selection
 - **git fr** : Git fetch and reset sources with interactive selection
 - **git ftr** : Git fetch with tags and reset sources with interactive selection
 - **git k** : Fetch all branches and show complete history in gitk
 - **git tig** : Fetch all branches and show complete history in tig
-

git log - Show commit logs

- **git l** : Show commits logs in oneline simple view without decorators
 - **git lo** : Show commits logs in oneline simple view
-

git merge - Merge development histories together

- **git mt** : Run merge conflicts resolution tools

git push - Push references and objects to remotes

- **git p** : Push references and objects to remotes
 - **git pf** : Force push references and objects to remotes
 - **git pu** : Git push with interactive selection
 - **git put** : Git push tags with interactive selection
-

git rebase - Reapply and rework commits history

- **git ra** : Abort current rebase operation and reset HEAD to the original state
 - **git rb** : Git rebase with interactive selection
 - **git rbl** : Git rebase local commits with interactive selection
 - **git rc** : Continue the rebasing process after merge conflict resolutions or commit amends
 - **git re** : Edit the todo list during an interactive rebase
 - **git rs** : Skip the current patch during a rebasing process
 - **git r** : Interactive rebase of the last N commits (default 5)
 - **git redit** : Edit through rebase the last N commits (default 5)
 - **git rf** : Interactive rebase from commit reference
 - **git rredit** : Edit through rebase from commit reference
-

git remote - Manage remote sources repositories

- **git rv** : List the repositories remotes with their URL

git reset - Reset history or sources to a specific state

- **git rhf** : Reset history and sources to the last fetched commit
 - **git rhh** : Reset history and sources to the current commit
 - **git ri** : Reset history only to the previous commit
 - **git ro** : Reset history and sources to the previous commit
 - **git rt** : Reset history and sources to a specific commit
 - **git revertf** : Revert changes of the specified commit (default: HEAD)
 - **git rl** : Revert the current commit interactively
-

git show - Show commit contents or other objects

- **git shall** : Show commit with all differences highlighted
 - **git shf** : Show commit with full author, commiter and dates details
 - **git shm** : Show commit with only names and status of changed files
-

git stash - Store or stash uncommitted changes away

- **git s** : Stash the changes in a dirty working directory away
- **git sc** : Remove all the stash entries
- **git sp** : Interactively select hunks to be stashed
- **git spop** : Extract last stash state to the current working tree
- **git cl** : Reset and stash changes
- **git cla** : Abort cherry-pick, abort am, reset and stash changes
- **git su** : Stash changes including untracked files

git submodule - Initialize, update or inspect submodules

- **git sm** : Show the status of the submodules
- **git smi** : Initialize the submodules recorded in the sources
- **git smu** : Synchronize, initialize and update submodules recursively
- **git clean-tags** : Clean all tags automatically

git-dct