git-dct

Author: Adrian DC

Description: Git development CLI tools for daily usage

Date: 10/02/2025

Version: 2.1.1

▸ Usage

Git development CLI tools for daily usage

Documentation: https://radiandevcore.gitlab.io/tools/git-dct
Package: https://pypi.org/project/git-dct/

Table of contents

Usage

```
usage: git-dct [-h] [--version] [--no-color] [--update-check] [--settings] [--set GROUP KEY VAL] [-e | -d] [--]

git-dct: Git development CLI tools for daily usage

internal arguments:
  -h, --help           # Show this help message
  --version            # Show the current version
  --no-color           # Disable colors outputs with 'NO_COLOR=1'
                       # (or default settings: [themes] > no_color)
  --update-check       # Check for newer package updates
  --settings           # Show the current settings path and contents
  --set GROUP KEY VAL  # Set settings specific 'VAL' value to [GROUP] > KEY
                       # or unset by using 'UNSET' as 'VAL'

modes arguments:
  -e, --enable         # Enable profiles configurations
  -d, --disable        # Disable profiles configurations

positional arguments:
  --                   # Positional arguments separator (recommended)
```

Enable Git configurations

```
git dct --enable
```

Disable Git configurations

```
git dct --disable
```

Userspace available settings

`git-dct` creates a `settings.ini` configuration file in a userspace folder.

For example, it allows to disable the automated updates daily check (`[updates] > enabled`)

The `settings.ini` file location and contents can be shown with the following command:

```
git-dct --settings
```

Environment available configurations

`git-dct` uses `colored` for colors outputs and `questionary` for interactive menus.

If colors of both outputs types do not match the terminal's theme,
an environment variable `NO_COLOR=1` can be defined to disable colors.

Dependencies

- colored: Terminal colors and styles
- questionary: Interactive terminal user interfaces
- setuptools: Build and manage Python packages
- update-checker: Check for package updates on PyPI

References

- commitizen: Simple commit conventions for internet citizens
- git-cliff: CHANGELOG generator
- gitlab-release: Utility for publishing on GitLab
- gcil: Launch .gitlab-ci.yml jobs locally
- mkdocs: Project documentation with Markdown
- mkdocs-exporter: Exporter plugin for mkdocs documentation
- mkdocs-material: Material theme for mkdocs documentation
- mypy: Optional static typing for Python
- pre-commit: A framework for managing and maintaining pre-commit hooks
- pre-commit-crocodile: Git hooks intended for developers using pre-commit
- PyPI: The Python Package Index
- twine: Utility for publishing on PyPI

git-dct

Author: Adrian DC

Description: Git development CLI tools for daily usage

Date: 10/02/2025

Version: 2.1.1

▸ Aliases

Git development CLI tools for daily usage

Table of contents

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 rfedit :** 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 uncomitted 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

Author: Adrian DC

Description: Git development CLI tools for daily usage

Date: 10/02/2025

Version: 2.1.1

▸ Tools

Git development CLI tools for daily usage

Table of contents

git cauthor - Update commit authorship from current user or a commit

```
usage: git cauthor [-h] [--validate] [from_commit]

git cauthor: Update commit authorship from current user or a commit (git-dct)

tool arguments:
  -h, --help   Show this help message
  --validate   Validate result by requesting user input
```

---

git fcp - Git fetch path and cherry-pick commits

```
usage: git fcp [-h] [--validate] [path] [commits]

git fcp: Git fetch path and cherry-pick commits (git-dct)

tool arguments:
  -h, --help  Show this help message
  --validate  Validate result by requesting user input
  path        Sources folder to fetch from
```

---

git fe - Git fetch with interactive selection

```
usage: git fe [-h] [--reset] [--tags] [--validate] [remote] [branch]

git fe: Git fetch and reset sources with interactive selection (git-dct)

tool arguments:
  -h, --help  Show this help message
  --reset     Reset sources to fetched remote branch
  --tags      Fetch and refresh all tags
  --validate  Validate result by requesting user input
  remote      Remote repository name (default: auto)
```

---

git pu - Git push with interactive selection

```
usage: git pu [-h] [-f] [-r REF] [-t] [--no-verify] [--validate]
              [remote] [branch]

git pu: Git push with interactive selection (git-dct)

tool arguments:
  -h, --help         Show this help message
  -f, --force        Force push changes
  -r REF, --ref REF  Reference to push (default: HEAD)
  -t, --tags         Push tags instead of branches
  --no-verify        Disable pre-commit 'pre-push' hooks
  --validate         Validate result by requesting user input
  remote             Remote repository name (default: auto)
```

git rb - Git rebase with interactive selection

```
usage: git rb [-h] [--local] [--validate] [remote] [branch]

git rb: Git rebase with interactive selection (git-dct)

tool arguments:
  -h, --help  Show this help message
  --local     Rebase only local commits against remote branch
  --validate  Validate result by requesting user input
  remote      Remote repository name (default: auto)
```

git stat - Git history with remote comparator

```
usage: git stat [-h] [-r REF] [--stats-only] [--validate] [remote] [branch]

git stat: Git history with remote comparator (git-dct)

tool arguments:
  -h, --help         Show this help message
  -r REF, --ref REF  Reference to push (default: HEAD)
  --stats-only       Show only stats of the differences
  --validate         Validate result by requesting user input
  remote             Remote repository name (default: auto)
```

git-dct