backlog by nulab + git

Backlog lets you manage
projects with Git integration

Learn more

Commit log operations

- Amend previous commit
- Amend only comments of previous commit
- Amend past commit
- Amend only comments of a past commit
- Exit a rebase
- View HEAD changes across history
- View changes in branch tip
- Remove previous commit
- Reset a rebase
- Cancel the previous reset
- Copy a commit from another branch
- Search for a commit that includes a specific comment

Amend previous commit

```
$ git commit --amend
```

Assign the --amend option to overwirte the latest commit of the branch you're wokring on.

Learn Git Basics: rewriting history

Work with Git: commit --amend

When there are no files in an index, you can recommit the previous commit by assigning the --amend option and you'll be prompted to edit/amend the existing commit comment.

Learn Git Basics: rewriting history

Work with Git: commit --amend

Amend past commit

```
$ git rebase -i <commit>
```

Assign a commit hash and a list of all commits up to the assigned commit will be listed. Find the commit you wish to amend, and change that line from "pick" to "edit", then save and quit.

Next, assign the --amend option to commit. A screen for adding comments will be displayed. Amend the comment.

```
$ git commit --amend
```

Finally, assign the --continue option to run rebase.

```
$ git rebase --continue
```

Learn Git Basics: rewriting history

Work with Git: change commit using rebase

Amend only comments of a past commit

```
$ git rebase -i <commit>
```

comments will be displayed. Amend the comment.

```
$ git commit --amend
```

Finally, assign the --continue option to run rebase.

```
$ git rebase --continue
```

Learn Git Basics: rewriting history

Work with Git: change commit using rebase

Exit a rebase

```
$ git rebase --abort
```

By adding the --abort option, you can exit the rebase operation.

View HEAD changes across history

```
$ git reflog
```

The reflog command allows you to see a list of commits that HEAD used to indicate in the past.

```
08084a5 HEAD@{0}: commit: append description of
the pull command
99daed2 HEAD@{1}: commit: append description of
the commit command
48eec1d HEAD@{2}: checkout: moving from master
to issue1
```

Both deleted and successful commits gathered by rebase will be displayed.

View changes in branch tip

```
$ git reflog <ref>
```

By assigning a branch name to , a list of commits that the tip of the branch used to indicate will be listed like below

```
445e0ae issue1@{0}: commit (merge): Merge branch
'master' into issue1
1c904bd issue1@{1}: commit (amend): modify
description of the pull command
08084a5 issue1@{2}: commit: append description
of the pull command
99daed2 issue1@{3}: commit: append description
of the commit command
48eec1d issue1@{4}: branch: Created from
48eec1ddf73a7fb508ef664efd6b3d873631742f
```

We can see the rebase history of both deleted and existing commits.

Remove previous commit

```
$ git reset --hard HEAD~
```

Learn Git Basics: undoing changes

Work with Git: git reset

Use the reflog command to look up commits before they were rebased. Identify the commit hash or the value of [HEAD@{number}] of the commit that occured before the rebase. In this example, 71bdfbd and HEAD@{4} are the references to the commit.

```
a51f8d2 HEAD@{0}: rebase -i (finish): returning
to refs/heads/dev
a51f8d2 HEAD@{1}: rebase -i (squash): update 1
3a273e1 HEAD@{2}: rebase -i (squash): updating
HEAD
f55ef69 HEAD@{3}: checkout: moving from dev to
f55ef69
71bdfbd HEAD@{4}: commit: update 2
f55ef69 HEAD@{5}: commit (amend): update 1
```

Assign the hash value (71bdfbd and HEAD@{4}) to <commit> when running the reset command.

The head position of the branch will now move to the commit before the rebase was executed. The state of the branch now will be identical to that before rebase was executed.

Cancel the previous reset

```
$ git reset --hard ORIG_HEAD
```

ORIG_HEAD refers to the commit before reset took place. You can revert the previous reset by assigning reset to ORIG_HEAD.

Learn Git Basics: undoing changes

Work with Git: git reset

The commit with the ID of <commit> will be copied to the current branch.

Learn Git Basics: rewriting history

Work with Git: cherry pick

Search for a commit that includes a specific comment

```
$ git log --grep "<pattern>"
```

Displays commits with text specified in <pattern>.

Prev Next

Backlog has built-in Git repositories for every project

Try it free

Follow us on

English Terms Privacy

© 2021 Nulab, Inc. All rights reserved.