


Proposed Additions to Stack Overflow

Images taken from Stack Overflow

Hiding Responses

 1554

```
git revert --no-commit 0766c053..HEAD
git commit
```

+50


This will revert everything from the HEAD back to the commit hash, meaning it will recreate that commit state in the working tree *as if* every commit since had been walked back. You can then commit the current tree, and it will create a brand new commit essentially equivalent to the commit you "reverted" to.

(The `--no-commit` flag lets git revert all the commits at once- otherwise you'll be prompted for a message for each commit in the range, littering your history with unnecessary new commits.)


This is a **safe and easy way to rollback to a previous state**. No history is destroyed, so it can be used for commits that have already been made public.

share improve this answer

edited Jun 28 '14 at 20:12

 user456814

answered Feb 12 '14 at 4:18

 Yarin

78.4k • 125 • 333 • 462

18

If you really do want to have individual commits (instead of reverting everything with one big commit), then you can pass `--no-edit` instead of `--no-commit`, so that you don't have to edit a commit message for each reversion. – user456814 Jun 28 '14 at 20:11


63

If one of the commits between 0766c053..HEAD is a merge then there will be an error popping up (to do with no -m specified). This may help those encountering that: stackoverflow.com/questions/5970889/ – timhc22 Nov 21 '14 at 11:55

Current

Hide

Ability to hide entire post

 1554

```
git revert --no-commit 0766c053..HEAD
git commit
```

+50


This will revert everything from the HEAD back to the commit hash, meaning it will recreate that commit state in the working tree *as if* every commit since had been walked back. You can then commit the current tree, and it will create a brand new commit essentially equivalent to the commit you "reverted" to.

(The `--no-commit` flag lets git revert all the commits at once- otherwise you'll be prompted for a message for each commit in the range, littering your history with unnecessary new commits.)


This is a **safe and easy way to rollback to a previous state**. No history is destroyed, so it can be used for commits that have already been made public.

share improve this answer

edited Jun 28 '14 at 20:12

 user456814

answered Feb 12 '14 at 4:18

 Yarin

78.4k • 125 • 333 • 462

Ability to hide/show individual comments

18

If you really do want to have individual commits (instead of reverting everything with one big commit), then you can pass `--no-edit` instead of `--no-commit`, so that you don't have to edit a commit message for each reversion. – user456814 Jun 28 '14 at 20:11

63

If one of the commits between 0766c053..HEAD is a merge then there will be an error popping up (to do with no -m specified). This may help those encountering that: stackoverflow.com/questions/5970889/ – timhc22 Nov 21 '14 at 11:55

Proposed

Hiding Responses

- The ability to hide comments and responses would allow users to condense pages considerably
- Would lessen the amount people are attacked for “gratuitous” responses
 - Users can hide responses, so no reason to call out others for it
- If top answer is not helpful for users, they can easily narrow down their focus on more helpful information more quickly

Specificity Track

41 Answers

active

oldest

votes

1

2

next



This depends a lot on what you mean by "revert".

9048

Temporarily switch to a different commit



If you want to temporarily go back to it, fool around, then come back to where you are, all you have to do is check out the desired commit:



```
# This will detach your HEAD, that is, leave you with no branch checked out:  
git checkout 0d1d7fc32
```

Or if you want to make commits while you're there, go ahead and make a new branch while you're at it:

```
git checkout -b old-state 0d1d7fc32
```

To go back to where you were, just check out the branch you were on again. (If you've made changes, as always when switching branches, you'll have to deal with them as appropriate. You could reset to throw them away; you could stash, checkout, stash pop to take them with you; you could commit them to a branch there if you want a branch there.)

Current

41 Answers

active

oldest

votes

1

2

next

detailed

moderate

simplified



This depends a lot on what you mean by "revert".

The level of detail can be selected by posters and toggled by users. By default, 'detailed' is selected

9048

Temporarily switch to a different commit



If you want to temporarily go back to it, fool around, then come back to where you are, all you have to do is check out the desired commit:



```
# This will detach your HEAD, that is, leave you with no branch checked out:  
git checkout 0d1d7fc32
```

Or if you want to make commits while you're there, go ahead and make a new branch while you're at it:

```
git checkout -b old-state 0d1d7fc32
```

To go back to where you were, just check out the branch you were on again. (If you've made changes, as always when switching branches, you'll have to deal with them as appropriate. You could reset to throw them away; you could stash, checkout, stash pop to take them with you; you could commit them to a branch there if you want a branch there.)

Proposed

Specificity Track

- Lowers threshold for users to contribute
- Still maintains quality of information users expect from stack overflow
- Provides varying levels of detail that are helpful for different users
- Additional means of filtering information and making page more concise