Fixing mistakes

Reuven M. Lerner, PhD
reuven@lerner.co.il • http://lerner.co.il/

Undoing mistakes

- One of the biggest reasons to use version control is in case you make a mistake
- So, if I've found an error in a file, how can I undo the mistake?

Option 1: New commit

- Sometimes it's just easiest to create a new commit with the fix
- Change the file, stage and commit the file, and the new commit will be just fine

Option 2: Revert

- This creates a new commit that reverts whatever was done in a previous commit.

```
git revert abcd
```

- You can revert any commit you want, not just the last one.

- This is another reason why it's better to have many small commits than one large one! (Reverting a small commit is easier…)

Revert!

```
$ git log --pretty=oneline
f7310fd5064c4cc649f96c9d12faedb8327fa296 Updated to say bye
a4d9e4f924765a1abec8a3ea711c657e8952aff5 Added .gitignore
b53dd549c4891f094ab427852899da958fb9ae21 I added a file

$ git revert f731
Waiting for Emacs...
[master 22d0108] Revert "Updated to say bye"
 1 files changed, 1 insertions(+), 2 deletions(-)

$ git log --pretty=oneline
22d010830febf7c85184908228e862fb17d2f7ca Revert "Updated to say bye"
f7310fd5064c4cc649f96c9d12faedb8327fa296 Updated to say bye
a4d9e4f924765a1abec8a3ea711c657e8952aff5 Added .gitignore
b53dd549c4891f094ab427852899da958fb9ae21 I added a file
```

So, what's a revert?

- A new commit! It builds on the last commit (as always), but this time it undoes that commit
- Reverts are commits like all others! They have a SHA1 of their own, and all of the same structure as other commits. We can even revert a revert…

What can you revert?

- You *can* revert any commit that you want. However, it's usually best to revert the most recent commit that changed a file.

- If you try to revert a commit other than the most recent one that changes a file, you'll be put in a merge state. (Type "git status", and you'll see.) In such a case, it's probably easiest to do a "git reset —hard" to get out of the merge.

Option 3: Amend

- If you forgot something in the previous commit, you can change it!

  ```
  git commit –amend
  ```

- The following also works, but **should not be used if you're using Gerrit**:

  ```
  git commit ––amend –m 'New message'
  ```

--amend thoughts

- If you enter a new commit message, it'll overwrite the existing one
- Everything from the previous commit is still in force (unless you changed it)
- Invisible to the outside world — so it lets you rewrite history!
- This is one of the things that people love and hate about Git, that history can be rewritten

Reverting far back

- If you try to revert an older commit, resulting in conflicts with the current commit, you'll basically end up having to merge.
- I suggest that you not do this!

Option 4: reset

- The “git reset” command has a number of uses
- One of them is to move both HEAD and the current branch to a particular commit