## Git: Walking back changes

**Staged (not yet committed or pushed)**
Unstage changes:

```
git reset HEAD -- .
```

Delete unstaged changes in working directory

```
git clean -df
git checkout -- .
```

**Committed (not yet pushed)**
Delete change if it was the last commit (index):

```
git reset --hard HEAD~1
```

Uncommit (but don't delete) :

```
git reset --soft HEAD~1
```

**Pushed files !**
("Revert" is the safest approach because it adds the resulting changes to a new commit. No history is destroyed - it just adds a commit on top of history - so it can be used for commits that have been made public)

Delete one commit. This will delete this one commit but keep subsequent commits (may be merge issues to be fixed):

```
git revert 66c053
```

Rollback to a previous state (i.e. delete multiple commits):

```
git revert --no-commit 66c053..HEAD
git commit
```

(Rolls back changes occurring after commit 66c053)

**What is the Git HEAD?** Tip of the branch you are working on

HEAD~1 is the last committed change HEAD~2 is two committed changes ago

## Git: Fixing a merge conflict and surviving VIM (when using revert)

**MERGE CONFLICTS**

**STEP 1:** View the merge conflict
(will download to the RStudio window):

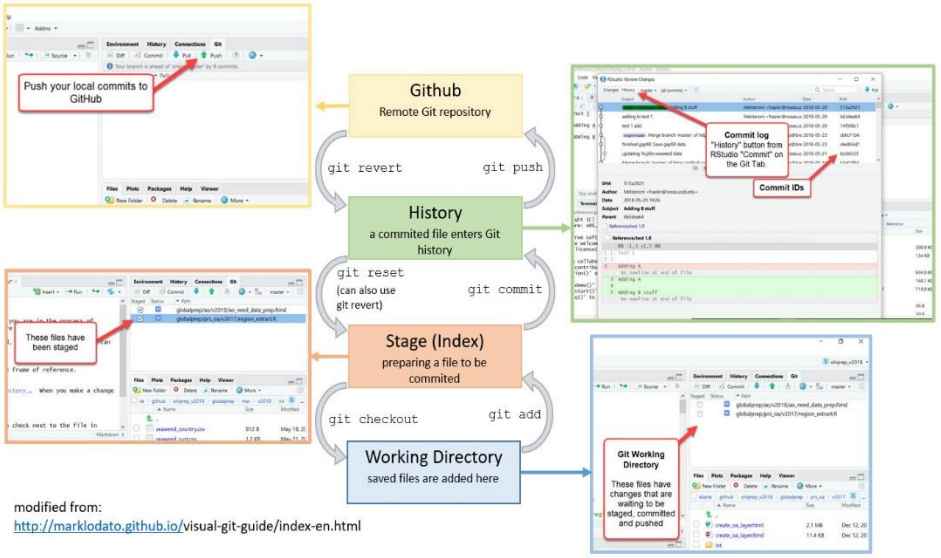
<<< HEAD
(portion of code on head, i.e, new addition)
====
(portion on last committed change, i.e., old version)
>>>>9d9c62
(the end of merge issue)

**STEP 2:** Manually fix and save
**STEP 3:** `git add --all`
**STEP 4:** `git commit -m "message"`
**STEP 5:** `git push`

**VIM**

**OPTION 1:** If you are happy with the automatically generated commit message (i.e., Revert "something something something"), quit VIM by typing:
`:q`

**OPTION 2:** If you want to alter the message, enter:
`i`
for insert mode, change the message and then:
Ctrl+C
`:qi`




modified from:
http://marklodato.github.io/visual-git-guide/index-en.html