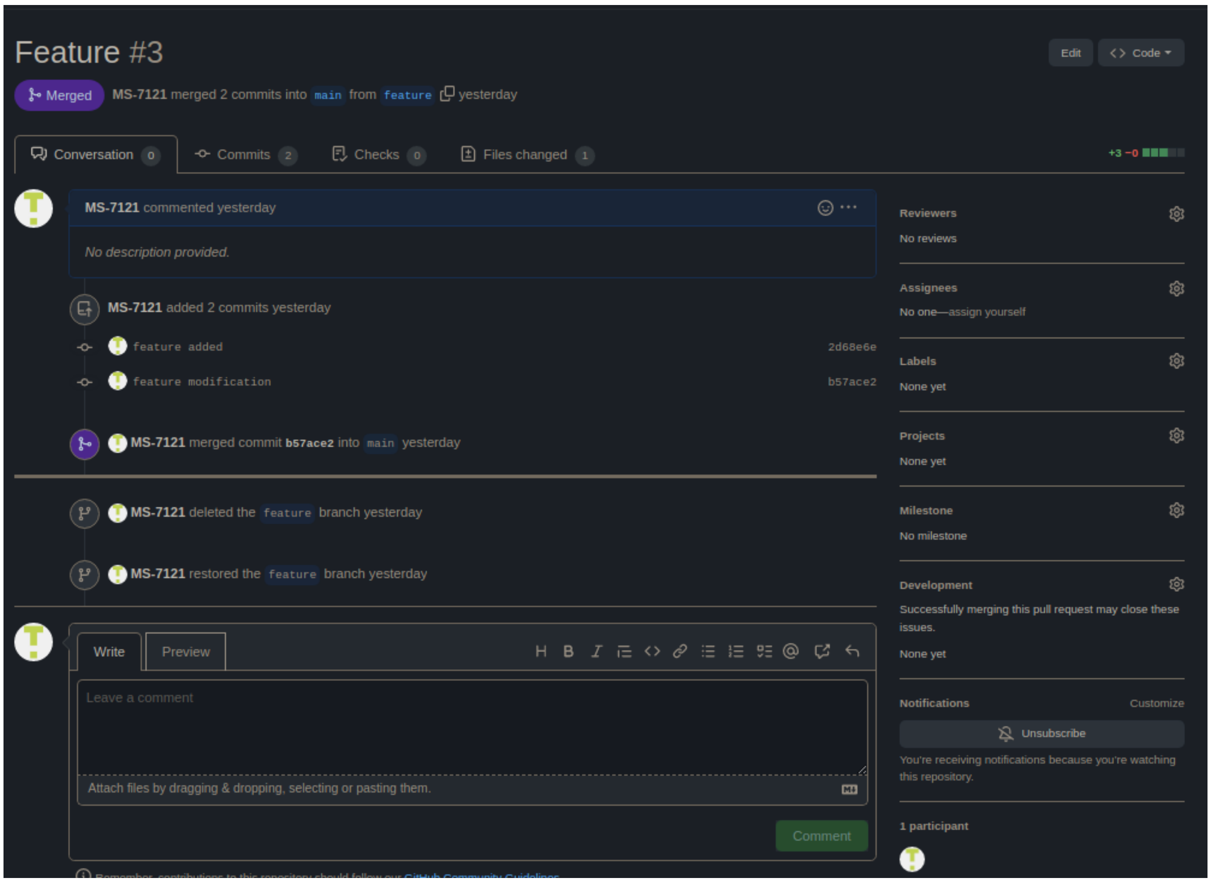
1) GIT Merge & Pull Request

- **GIT PULL REQUEST**

1. Let's first make changes in the Feature Branch.
2. After making commits we will push the changes to the git repository.
3. Here we will be able to see the Pull Request on the dashboard.
4. Later we will compare and create a pull request.
5. We shall resolve any merge conflicts created because of the merge and after that we should merge the branch.
6. After successful merging we will be able to see following details on the closed pull request section on the repo.



Feature #3

Merged MS-7121 merged 2 commits into main from feature yesterday

Conversation 0 · Commits 2 · Checks 0 · Files changed 1 · +3 −0

MS-7121 commented yesterday

No description provided.

MS-7121 added 2 commits yesterday

feature added 2d68e6e
feature modification b57ace2

MS-7121 merged commit b57ace2 into main yesterday

MS-7121 deleted the feature branch yesterday

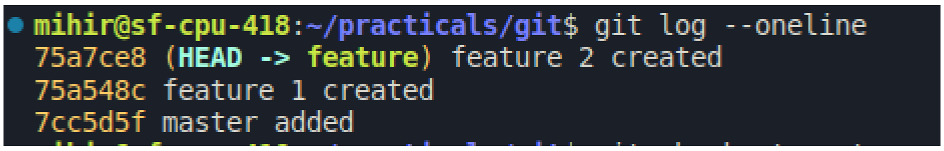
MS-7121 restored the feature branch yesterday

Reviewers: No reviews
Assignees: No one—assign yourself
Labels: None yet
Projects: None yet
Milestone: No milestone
Development: Successfully merging this pull request may close these issues. None yet
Notifications: Customize. Unsubscribe. You're receiving notifications because you're watching this repository.
1 participant

- **GIT MERGE**

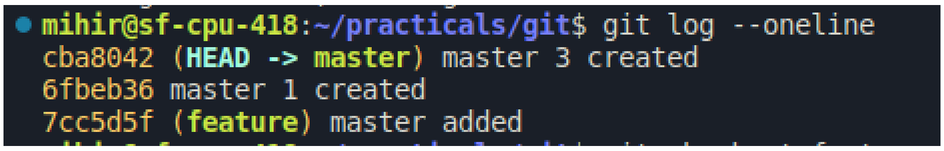
1. Let's create a branch named feature.
2. Make some commits onto the feature branch.
3. After that checkout to the master branch.
4. Use git merge command to merge with the feature branch.

Feature Branch:



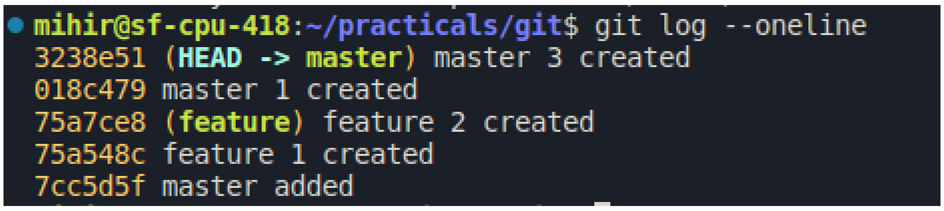
```
mihir@sf-cpu-418:~/practicals/git$ git log --oneline
75a7ce8 (HEAD -> feature) feature 2 created
75a548c feature 1 created
7cc5d5f master added
```

Main Branch:



```
mihir@sf-cpu-418:~/practicals/git$ git log --oneline
cba8042 (HEAD -> master) master 3 created
6fbeb36 master 1 created
7cc5d5f (feature) master added
```

Main Branch after merging with the Feature Branch:

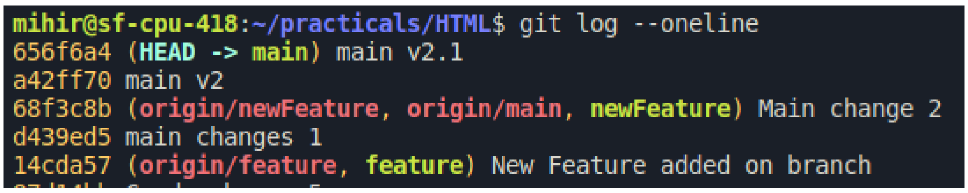


```
mihir@sf-cpu-418:~/practicals/git$ git log --oneline
3238e51 (HEAD -> master) master 3 created
018c479 master 1 created
75a7ce8 (feature) feature 2 created
75a548c feature 1 created
7cc5d5f master added
```

2) Rebase

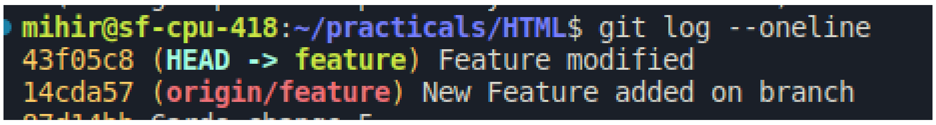
1. Let’s continue with the branch Feature.
2. Make changes and commits onto the Feature branch.
3. Checkout to the master branch.
4. Use command “git rebase feature”
5. Resolve any merge conflicts.
6. Use command “git rebase --continue" to complete the rebase.

Main Branch log before rebasing:



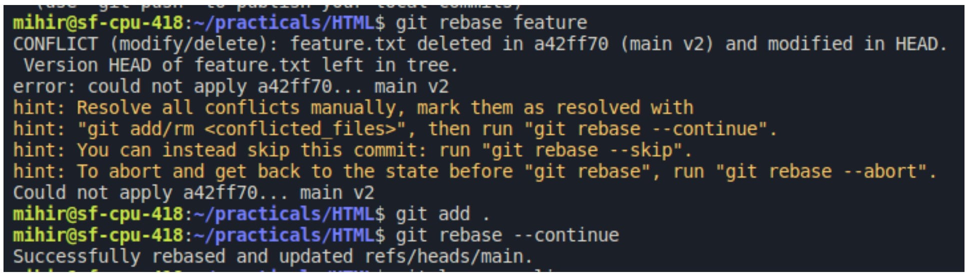
```
mihir@sf-cpu-418:~/practicals/HTML$ git log --oneline
656f6a4 (HEAD -> main) main v2.1
a42ff70 main v2
68f3c8b (origin/newFeature, origin/main, newFeature) Main change 2
d439ed5 main changes 1
14cda57 (origin/feature, feature) New Feature added on branch
```

Feature Branch log:



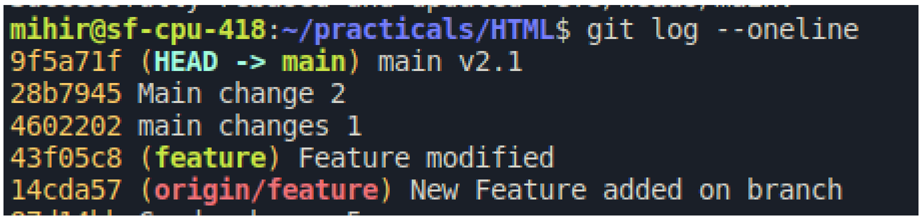
```
mihir@sf-cpu-418:~/practicals/HTML$ git log --oneline
43f05c8 (HEAD -> feature) Feature modified
14cda57 (origin/feature) New Feature added on branch
```

Rebase command:



```
mihir@sf-cpu-418:~/practicals/HTML$ git rebase feature
CONFLICT (modify/delete): feature.txt deleted in a42ff70 (main v2) and modified in HEAD.
 Version HEAD of feature.txt left in tree.
error: could not apply a42ff70... main v2
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git rebase --abort".
Could not apply a42ff70... main v2
mihir@sf-cpu-418:~/practicals/HTML$ git add .
mihir@sf-cpu-418:~/practicals/HTML$ git rebase --continue
Successfully rebased and updated refs/heads/main.
```

Main Branch log after rebasing:



```
mihir@sf-cpu-418:~/practicals/HTML$ git log --oneline
9f5a71f (HEAD -> main) main v2.1
28b7945 Main change 2
4602202 main changes 1
43f05c8 (feature) Feature modified
14cda57 (origin/feature) New Feature added on branch
```

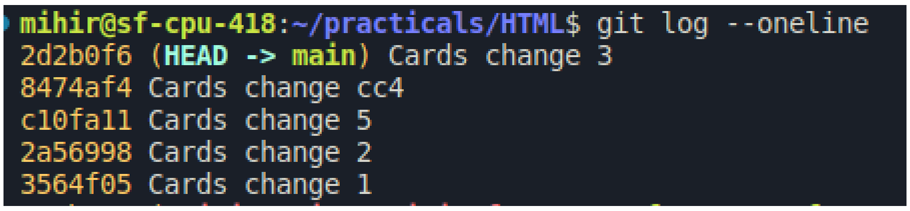
3) Change commit message

There are several ways in which we can change the commit message.

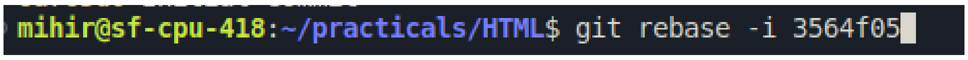
USING GIT REBASE (Interactive)

1. Enlist all the commits using the "git rebase -i <*hash value*>"
2. A dialog box will open containing all the commits.
3. Change "pick" to *"reword"* to change the commit message on particular commit line
4. A new editor will be prompt to change the message
5. Save and exit the editor.
6. The message will be changed.

Let's change commit message of 8474af4:

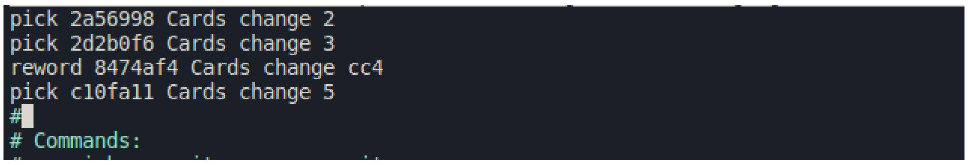


```
mihir@sf-cpu-418:~/practicals/HTML$ git log --oneline
2d2b0f6 (HEAD -> main) Cards change 3
8474af4 Cards change cc4
c10fa11 Cards change 5
2a56998 Cards change 2
3564f05 Cards change 1
```



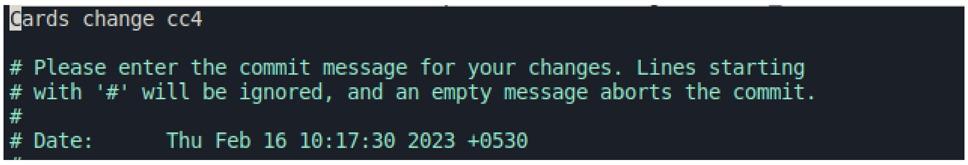
```
mihir@sf-cpu-418:~/practicals/HTML$ git rebase -i 3564f05
```

All the commits after particular hash value will be shown:



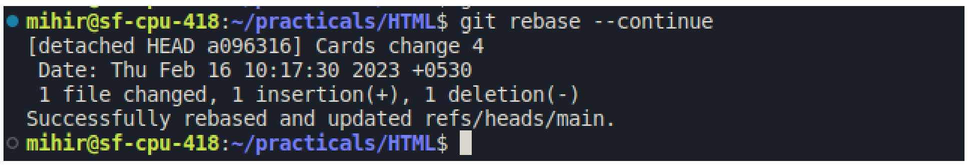
```
pick 2a56998 Cards change 2
pick 2d2b0f6 Cards change 3
reword 8474af4 Cards change cc4
pick c10fa11 Cards change 5
#
# Commands:
```

Editor prompt for changing the message:



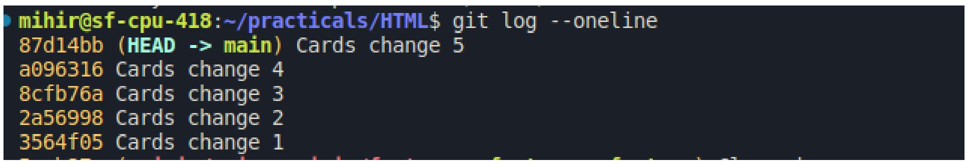
```
Cards change cc4

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Date:      Thu Feb 16 10:17:30 2023 +0530
```

Continue the rebase:



```
mihir@sf-cpu-418:~/practicals/HTML$ git rebase --continue
[detached HEAD a096316] Cards change 4
 Date: Thu Feb 16 10:17:30 2023 +0530
 1 file changed, 1 insertion(+), 1 deletion(-)
Successfully rebased and updated refs/heads/main.
mihir@sf-cpu-418:~/practicals/HTML$
```

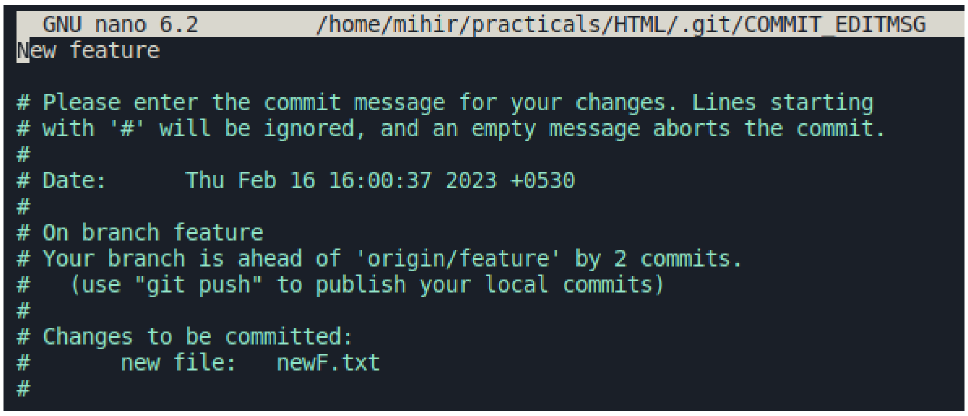
Final check:



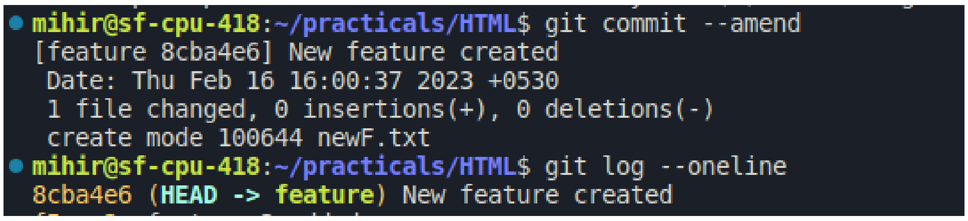
```
mihir@sf-cpu-418:~/practicals/HTML$ git log --oneline
87d14bb (HEAD -> main) Cards change 5
a096316 Cards change 4
8cfb76a Cards change 3
2a56998 Cards change 2
3564f05 Cards change 1
```

USING GIT AMMEND

1. Checkout to the branch that contains the commit.
2. Type “git commit –amend".
3. Editor will be opened, change the message and save.
4. The message will be updated.



```
  GNU nano 6.2          /home/mihir/practicals/HTML/.git/COMMIT_EDITMSG
New feature

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Date:      Thu Feb 16 16:00:37 2023 +0530
#
# On branch feature
# Your branch is ahead of 'origin/feature' by 2 commits.
#   (use "git push" to publish your local commits)
#
# Changes to be committed:
#       new file:   newF.txt
#
```

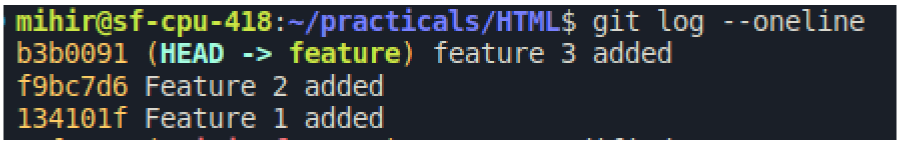


```
● mihir@sf-cpu-418:~/practicals/HTML$ git commit --amend
  [feature 8cba4e6] New feature created
   Date: Thu Feb 16 16:00:37 2023 +0530
   1 file changed, 0 insertions(+), 0 deletions(-)
   create mode 100644 newF.txt
● mihir@sf-cpu-418:~/practicals/HTML$ git log --oneline
  8cba4e6 (HEAD -> feature) New feature created
```

4) Cherry-pick

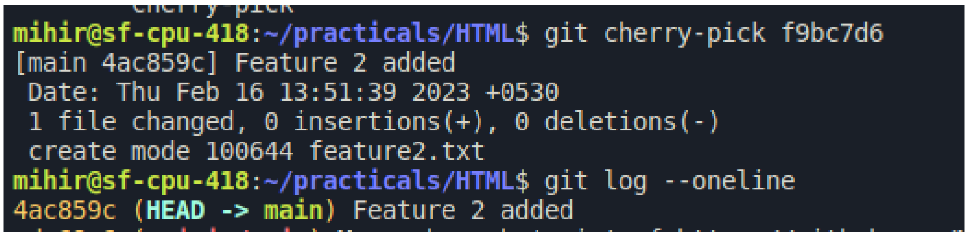
1. Let’s continue with the Feature branch.
2. Do multiple commits onto the feature branch.
3. Select particular hash of the branch which you want to add to the master branch.
4. Use “git cherry-pick *<hash value>”.*

Listing all the commits:



```
mihir@sf-cpu-418:~/practicals/HTML$ git log --oneline
b3b0091 (HEAD -> feature) feature 3 added
f9bc7d6 Feature 2 added
134101f Feature 1 added
```

Switching to the master branch and adding one of the commits onto it:

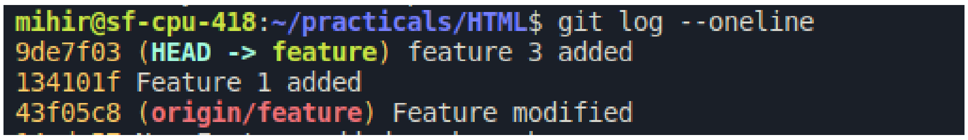


```
mihir@sf-cpu-418:~/practicals/HTML$ git cherry-pick f9bc7d6
[main 4ac859c] Feature 2 added
 Date: Thu Feb 16 13:51:39 2023 +0530
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 feature2.txt
mihir@sf-cpu-418:~/practicals/HTML$ git log --oneline
4ac859c (HEAD -> main) Feature 2 added
```

5) Drop commit

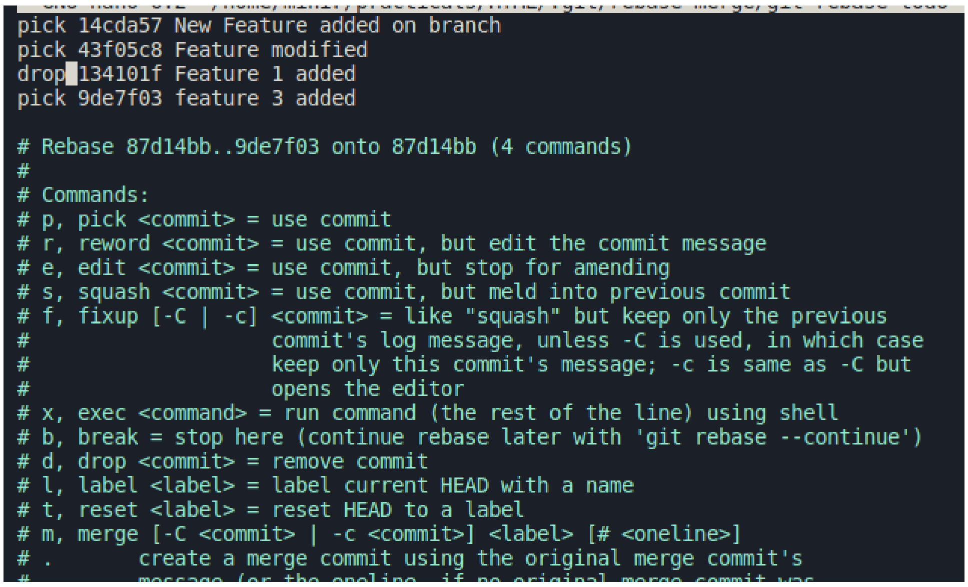
1. We will use git rebase for the dropping of the commit
2. Use “git rebase -i <*hash value*>”.
3. Change “pick” to “drop” for the commit we want to drop.

Log of all the commits:



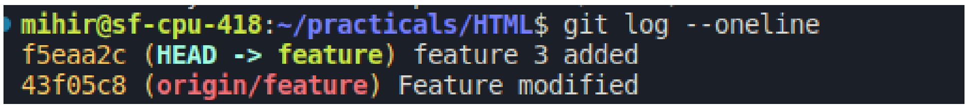
```
mihir@sf-cpu-418:~/practicals/HTML$ git log --oneline
9de7f03 (HEAD -> feature) feature 3 added
134101f Feature 1 added
43f05c8 (origin/feature) Feature modified
```

Selecting commit to drop:



```
pick 14cda57 New Feature added on branch
pick 43f05c8 Feature modified
drop 134101f Feature 1 added
pick 9de7f03 feature 3 added

# Rebase 87d14bb..9de7f03 onto 87d14bb (4 commands)
#
# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup [-C | -c] <commit> = like "squash" but keep only the previous
#                    commit's log message, unless -C is used, in which case
#                    keep only this commit's message; -c is same as -C but
#                    opens the editor
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
# .       create a merge commit using the original merge commit's
```

After dropping the commit:



```
mihir@sf-cpu-418:~/practicals/HTML$ git log --oneline
f5eaa2c (HEAD -> feature) feature 3 added
43f05c8 (origin/feature) Feature modified
```