GIT: From Beginner To Fearless

Squash And Merge Activity: Squashing commits during merge at GitHub

Brian Gorman, Author/Instructor/Trainer

Introduction

Many times we have a pull request that contains a feature that was developed in a local repository. The feature has been completed and has been through all the rigor required to be implemented. During the development of the feature, multiple commits were recorded, and the history at master would reflect each of these commits if the pull request is directly merged with a regular merge operation.

Depending on the rules of your repository, the size of the commit chain, and a number of personal factors, one option that can be done is to merge the pull request using a "Squash and Merge" operation. If this option is selected, the multiple commit history will be compacted into one commit at the time the merge is completed, with a regular merge message and details that automatically contain all of the commit messages [at GitHub].

An important word about using squash and merge, however, before everyone jumps on the 'this is incredibly awesome' bandwagon. When a squash and merge operation is completed, it is very critical that the feature branch is then deleted following the merge, both at Remote and at Local. Failing to do this will result in a commit history mismatch.

Depending on the order you've worked through some of the activities, you might have heard me talk about never changing history on a publicly available branch. This is the same thing in reverse, with the caveat that it is entirely possible to continue working on the feature branch at local, and the problem will mostly surface during merge when it looks like many commits need to be merged, even though their code bits should already be merged into master from the previous merge operation.

In this activity, we'll walk through doing the pull request with a squash and merge, and see what it looks like when we don't delete the branch, and then we'll do it again while also deleting the branch to show how I would recommend using this option for merging code.

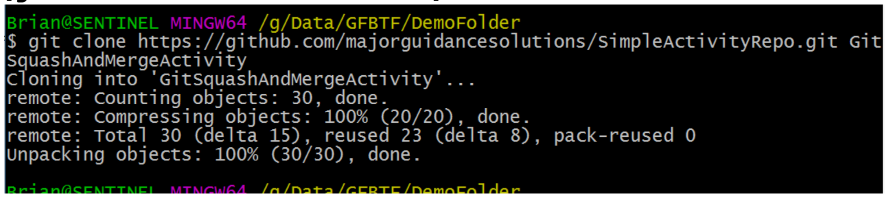
Let's gets started!

Step 1: Make sure you have a working repository that is up to date.

a) Start with any repo, make sure you have the latest in master, and create a feature branch.

First clone the repo if it doesn't exist:

[**`git clone <link> <folder>`**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder
$ git clone https://github.com/majorguidancesolutions/SimpleActivityRepo.git Git
SquashAndMergeActivity
Cloning into 'GitSquashAndMergeActivity'...
remote: Counting objects: 30, done.
remote: Compressing objects: 100% (20/20), done.
remote: Total 30 (delta 15), reused 23 (delta 8), pack-reused 0
Unpacking objects: 100% (30/30), done.

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder
```

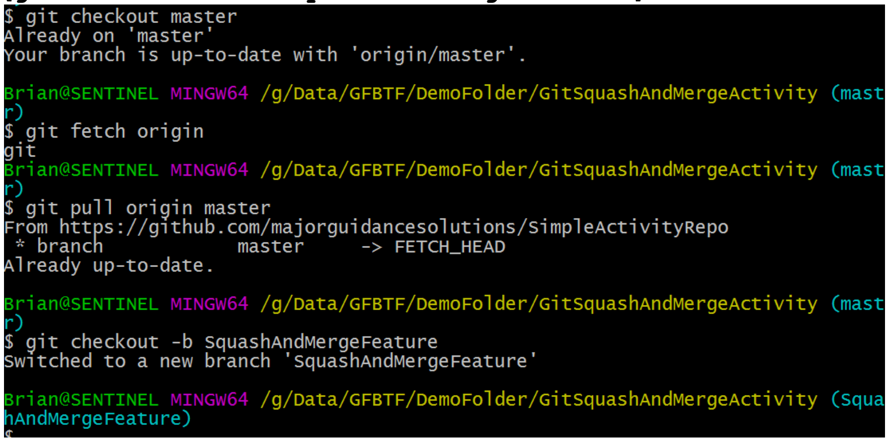
If you didn't clone, make sure master is up to date

[**`git checkout master`**]

[**`git fetch origin`**]

[**`git pull origin master`**]

[**`git checkout -b SquashAndMergeFeature`**]



```
$ git checkout master
Already on 'master'
Your branch is up-to-date with 'origin/master'.

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (mast
r)
$ git fetch origin
git
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (mast
r)
$ git pull origin master
From https://github.com/majorguidancesolutions/SimpleActivityRepo
 * branch            master     -> FETCH_HEAD
Already up-to-date.

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (mast
r)
$ git checkout -b SquashAndMergeFeature
Switched to a new branch 'SquashAndMergeFeature'

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (Squa
hAndMergeFeature)
$
```

Step 2: Make four commits, push, squash and merge.

a) For this activity, we need to do 3-4 commits on our branch.

[**`code info.txt`**] //leave it open after saving

[**`git commit -am "Squash and Merge commit #1"`**]

[make another change in info.txt]

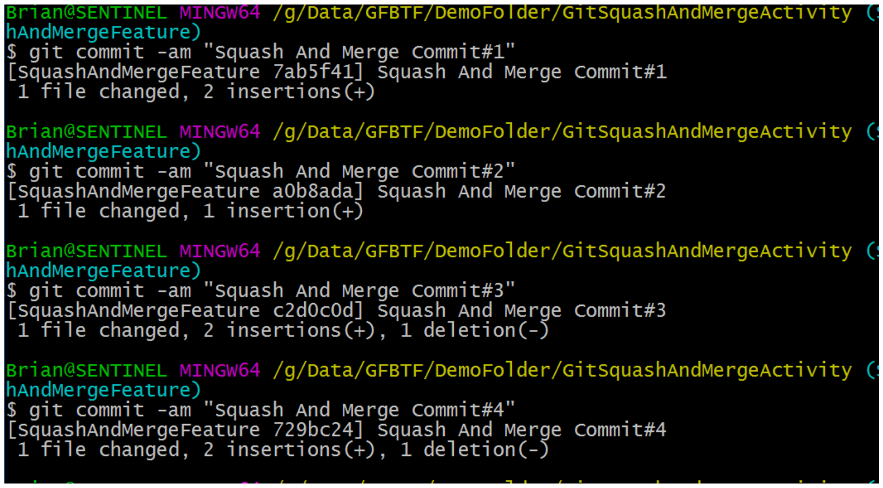
[**`git commit -am "Squash and Merge commit #2"`**]

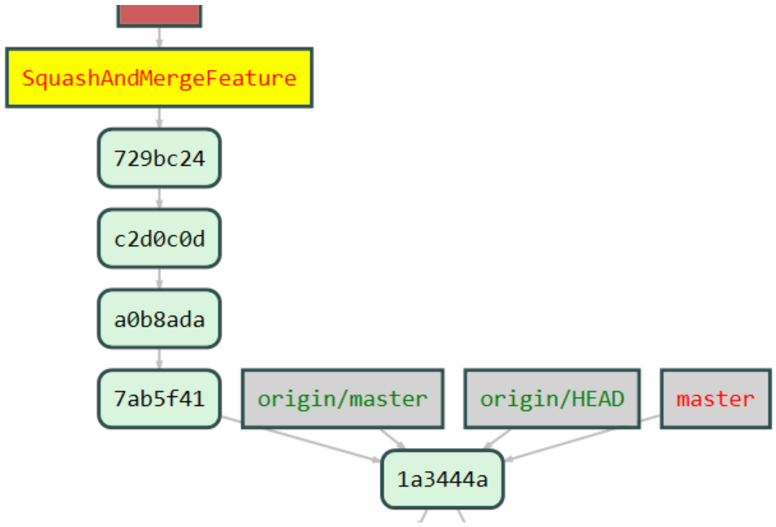
[make another change in info.txt]

[**`git commit -am "Squash and Merge commit #3"`**]

Notes

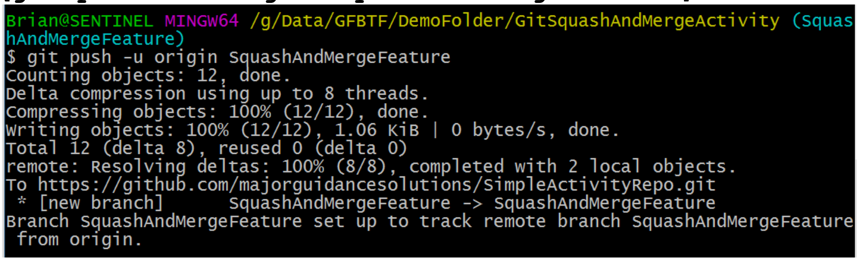
[make another change in info.txt]
[**git commit -am "Squash and Merge commit #4"**]

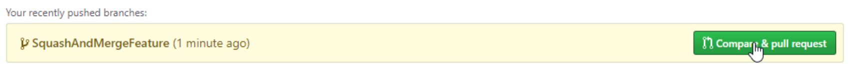

```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (S
hAndMergeFeature)
$ git commit -am "Squash And Merge Commit#1"
[SquashAndMergeFeature 7ab5f41] Squash And Merge Commit#1
 1 file changed, 2 insertions(+)

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (S
hAndMergeFeature)
$ git commit -am "Squash And Merge Commit#2"
[SquashAndMergeFeature a0b8ada] Squash And Merge Commit#2
 1 file changed, 1 insertion(+)

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (S
hAndMergeFeature)
$ git commit -am "Squash And Merge Commit#3"
[SquashAndMergeFeature c2d0c0d] Squash And Merge Commit#3
 1 file changed, 2 insertions(+), 1 deletion(-)

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (S
hAndMergeFeature)
$ git commit -am "Squash And Merge Commit#4"
[SquashAndMergeFeature 729bc24] Squash And Merge Commit#4
 1 file changed, 2 insertions(+), 1 deletion(-)
```

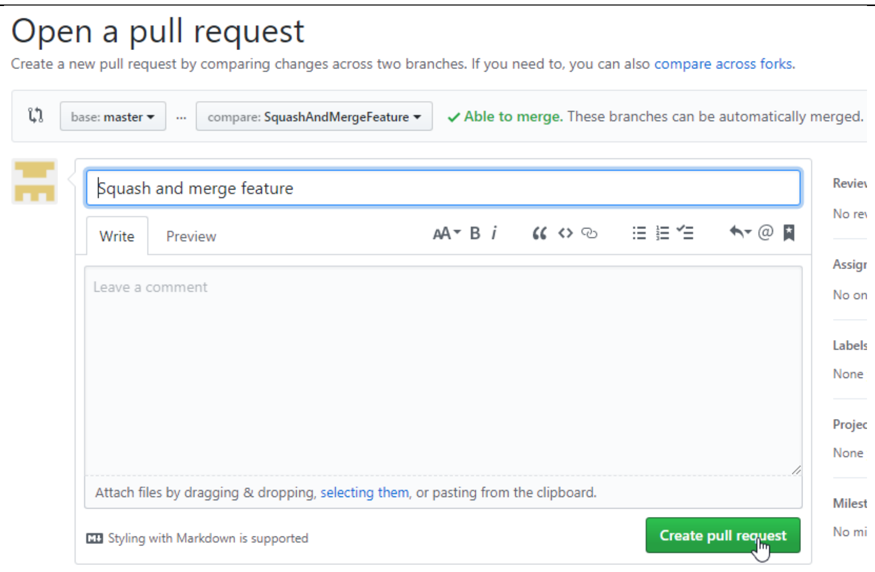


b) Push to GitHub, Create a Pull Request

[**git push -u origin SquashAndMergeFeature**]


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (Squas
hAndMergeFeature)
$ git push -u origin SquashAndMergeFeature
Counting objects: 12, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (12/12), 1.06 KiB | 0 bytes/s, done.
Total 12 (delta 8), reused 0 (delta 0)
remote: Resolving deltas: 100% (8/8), completed with 2 local objects.
To https://github.com/majorguidancesolutions/SimpleActivityRepo.git
 * [new branch]      SquashAndMergeFeature -> SquashAndMergeFeature
Branch SquashAndMergeFeature set up to track remote branch SquashAndMergeFeature
 from origin.
```


Your recently pushed branches:
SquashAndMergeFeature (1 minute ago)


Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

base: master ... compare: SquashAndMergeFeature ✓ Able to merge. These branches can be automatically merged.

Squash and merge feature

Write Preview

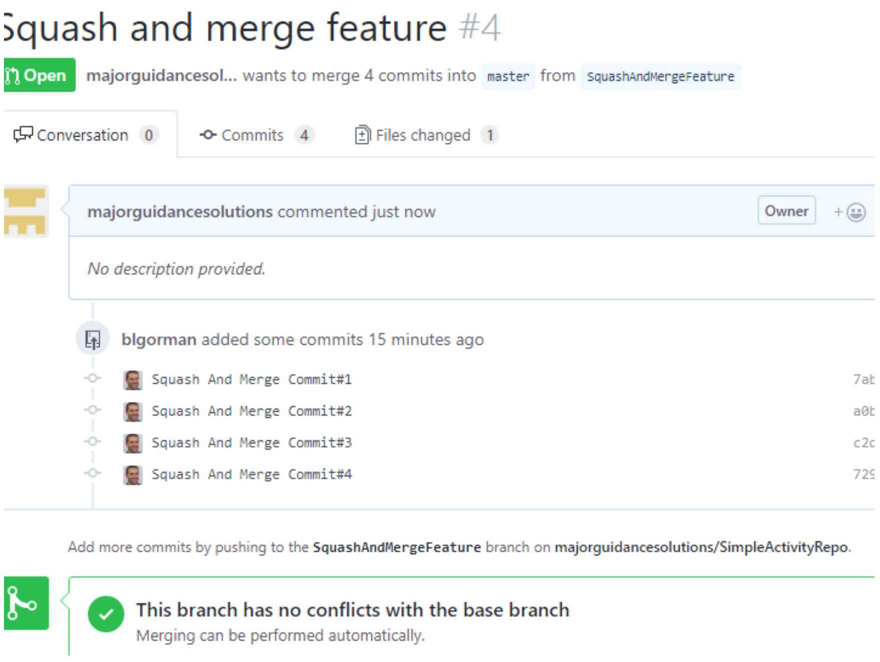
Leave a comment

Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

Styling with Markdown is supported

Create pull request

Reviev
No rev
Assigr
No on
Labels
None
Projec
None
Milest
No mi


Squash and merge feature #4

Open majorguidancesol... wants to merge 4 commits into master from SquashAndMergeFeature

Conversation 0 Commits 4 Files changed 1

majorguidancesolutions commented just now Owner

No description provided.

blgorman added some commits 15 minutes ago

```
Squash And Merge Commit#1                                  7at
Squash And Merge Commit#2                                  a0t
Squash And Merge Commit#3                                  c2c
Squash And Merge Commit#4                                  72g
```

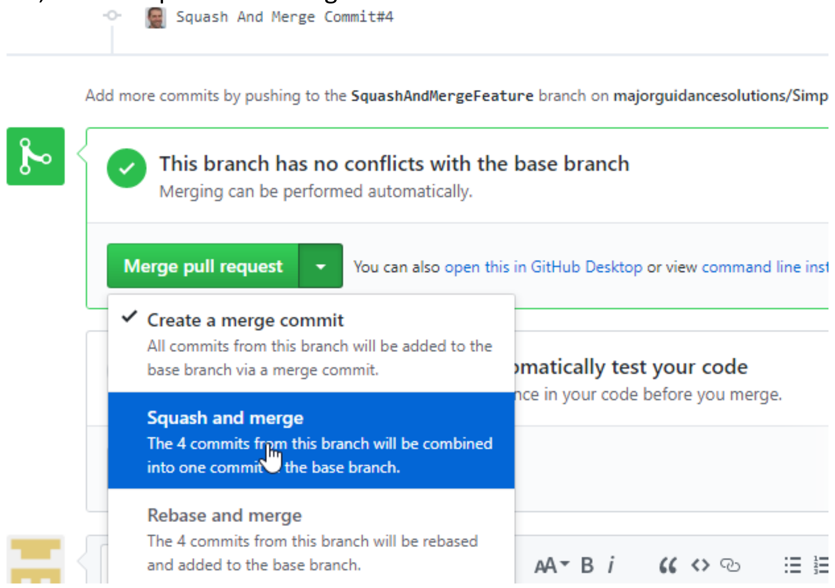
Add more commits by pushing to the **SquashAndMergeFeature** branch on **majorguidancesolutions/SimpleActivityRepo**.

This branch has no conflicts with the base branch

Merging can be performed automatically.

c) Squash and merge the request – DO NOT delete branch

First, select "Squash And Merge"


Squash And Merge Commit#4

Add more commits by pushing to the **SquashAndMergeFeature** branch on **majorguidancesolutions/Simp**

This branch has no conflicts with the base branch
Merging can be performed automatically.

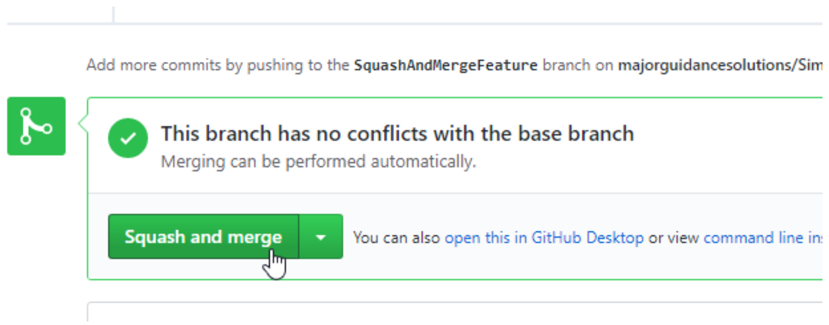
Merge pull request
You can also open this in GitHub Desktop or view command line inst

✓ Create a merge commit
All commits from this branch will be added to the base branch via a merge commit.

Squash and merge
The 4 commits from this branch will be combined into one commit in the base branch.

Rebase and merge
The 4 commits from this branch will be rebased and added to the base branch.

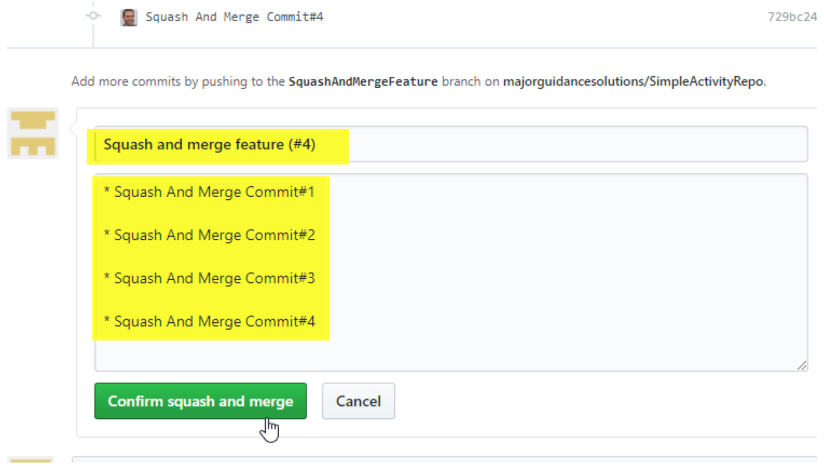
Next, hit the 'Squash and Merge' button


Add more commits by pushing to the **SquashAndMergeFeature** branch on **majorguidancesolutions/Sim**

This branch has no conflicts with the base branch
Merging can be performed automatically.

Squash and merge
You can also open this in GitHub Desktop or view command line in

Then note the commit messages are put into the details. You can change the commit message if you would like

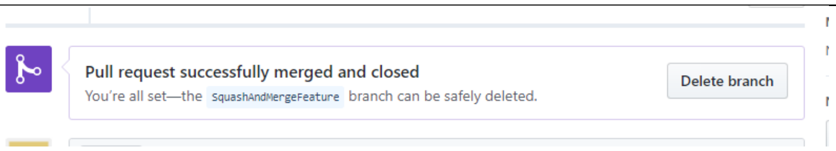

Squash And Merge Commit#4 729bc24

Add more commits by pushing to the **SquashAndMergeFeature** branch on **majorguidancesolutions/SimpleActivityRepo**.

Squash and merge feature (#4)

* Squash And Merge Commit#1
* Squash And Merge Commit#2
* Squash And Merge Commit#3
* Squash And Merge Commit#4

Confirm the squash and merge [reminder, do not delete the branch]


Pull request successfully merged and closed

You're all set—the SquashAndMergeFeature branch can be safely deleted.

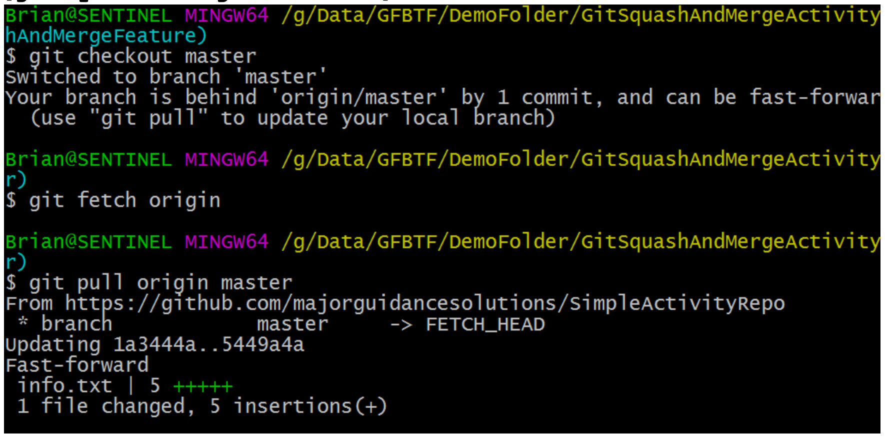
Step 3: Go back to local and get master up to date, then compare with the feature branch.

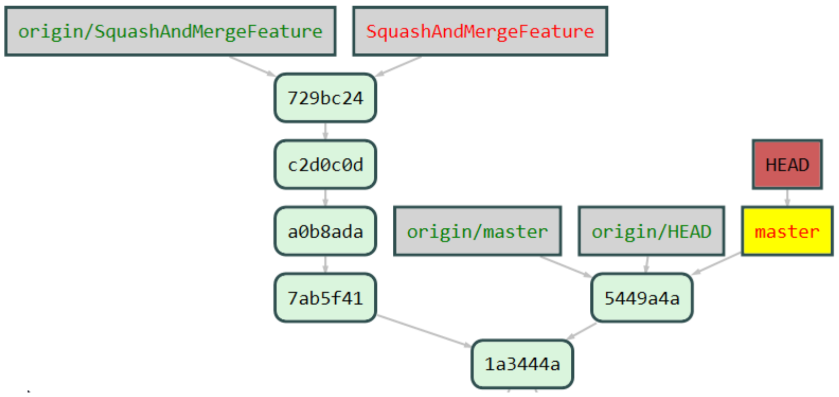
a) Get LOCAL master up to date

[git checkout master]

[git fetch origin]

[git pull origin master]


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity
hAndMergeFeature)
$ git checkout master
Switched to branch 'master'
Your branch is behind 'origin/master' by 1 commit, and can be fast-forwar
  (use "git pull" to update your local branch)

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity
r)
$ git fetch origin

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity
r)
$ git pull origin master
From https://github.com/majorguidancesolutions/SimpleActivityRepo
 * branch            master     -> FETCH_HEAD
Updating 1a3444a..5449a4a
Fast-forward
 info.txt | 5 +++++
 1 file changed, 5 insertions(+)
```

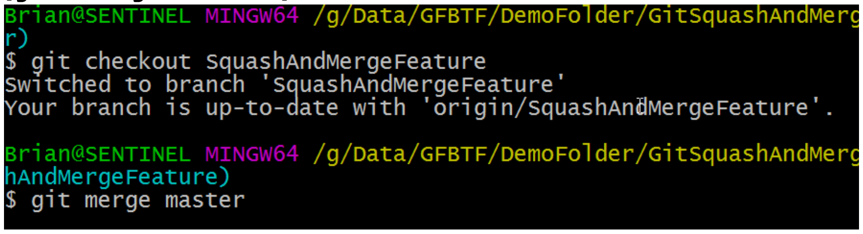


Here we can see that our feature four commits do not line up with the master commit history – this is to be expected, but it poses a problem. If we were to try to do a pull request we end up looking like we have multiple commits.

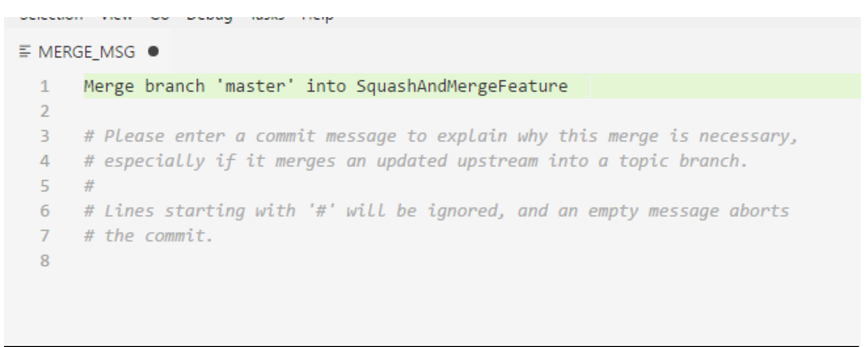
b) Merge master into feature

[**`git checkout <feature>`**]

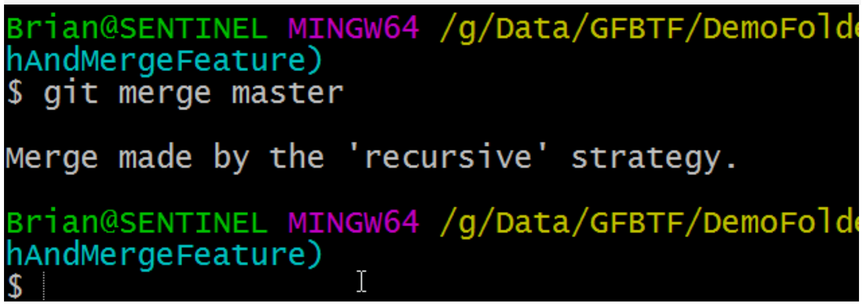
[**`git merge master`**]



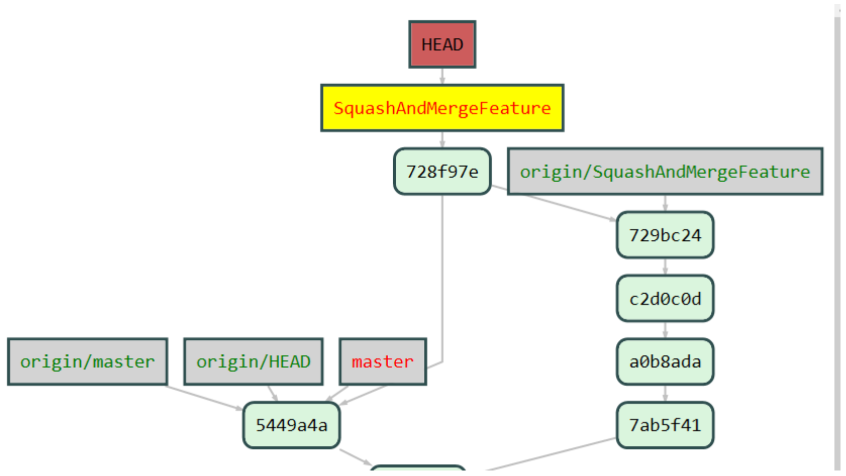
```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMerg
r)
$ git checkout SquashAndMergeFeature
Switched to branch 'SquashAndMergeFeature'
Your branch is up-to-date with 'origin/SquashAndMergeFeature'.

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMerg
hAndMergeFeature)
$ git merge master
```



```
MERGE_MSG ●
Merge branch 'master' into SquashAndMergeFeature

# Please enter a commit message to explain why this merge is necessary,
# especially if it merges an updated upstream into a topic branch.
#
# Lines starting with '#' will be ignored, and an empty message aborts
# the commit.
```



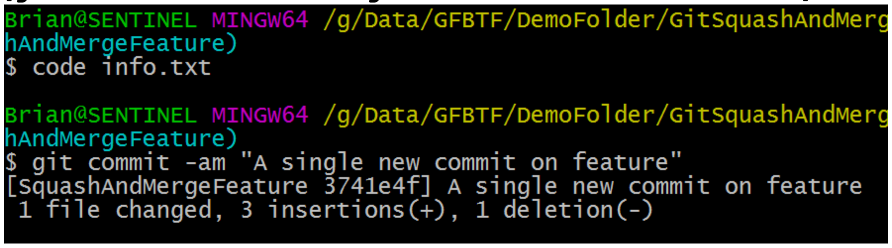
```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolde
hAndMergeFeature)
$ git merge master

Merge made by the 'recursive' strategy.

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolde
hAndMergeFeature)
$
```



So now our local master has nothing in it that feature doesn't. Also, the original four commits are in master as one commit. Let's add one quick change to the feature.

[**`code info.txt`**]

[git commit -am "A single new commit on feature"]

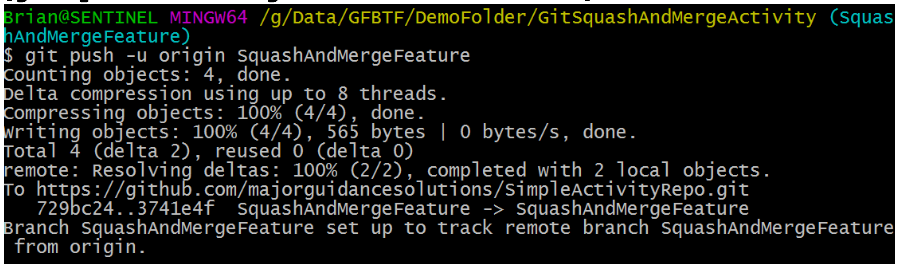


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMerg
hAndMergeFeature)
$ code info.txt

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMerg
hAndMergeFeature)
$ git commit -am "A single new commit on feature"
[SquashAndMergeFeature 3741e4f] A single new commit on feature
 1 file changed, 3 insertions(+), 1 deletion(-)
```

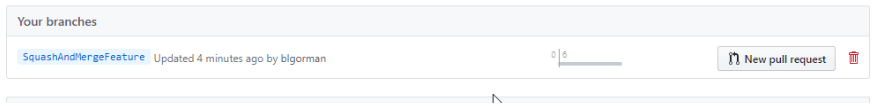
Step 3: Push to GitHub and merge.

a) Now let's do another push and create a pull request at GitHub to see what this looks like…

[git push -u origin <featurebranch>]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (Squas
hAndMergeFeature)
$ git push -u origin SquashAndMergeFeature
Counting objects: 4, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 565 bytes | 0 bytes/s, done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/majorguidancesolutions/SimpleActivityRepo.git
   729bc24..3741e4f  SquashAndMergeFeature -> SquashAndMergeFeature
Branch SquashAndMergeFeature set up to track remote branch SquashAndMergeFeature
 from origin.
```

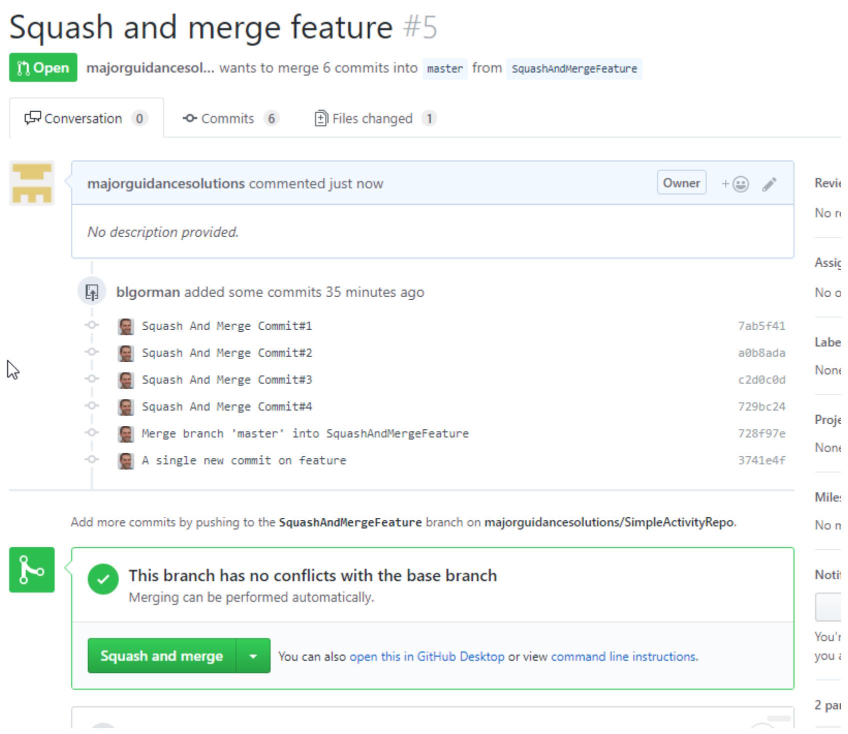


Your branches

SquashAndMergeFeature Updated 4 minutes ago by blgorman 0 | 6

6 commits ahead. Obviously there are only the two we would want (merge master and the new change). This shows why not deleting the branch is an issue. What if this was a major change? Would you 'trust' that your original changes were in master?

Create the pull request. Before merging, look:



Squash and merge feature #5

Open majorguidancesol... wants to merge 6 commits into `master` from `SquashAndMergeFeature`

Conversation 0 | Commits 6 | Files changed 1

majorguidancesolutions commented just now — Owner

No description provided.

blgorman added some commits 35 minutes ago

```
Squash And Merge Commit#1                              7ab5f41
Squash And Merge Commit#2                              a0b8ada
Squash And Merge Commit#3                              c2d0c0d
Squash And Merge Commit#4                              729bc24
Merge branch 'master' into SquashAndMergeFeature       728f97e
A single new commit on feature                         3741e4f
```

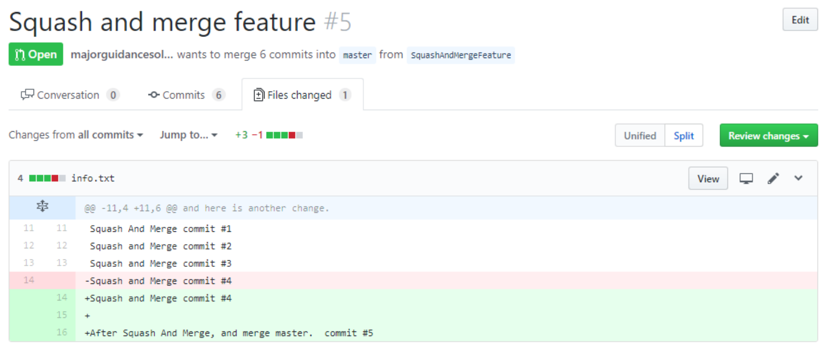
Add more commits by pushing to the **SquashAndMergeFeature** branch on **majorguidancesolutions/SimpleActivityRepo**.

This branch has no conflicts with the base branch
Merging can be performed automatically.

Squash and merge — You can also open this in GitHub Desktop or view command line instructions.

There are my first four commits again, the merge commit, and the new change. 6 commits to get up to date for a simple change.
Notice also that the squash and merge option is still selected. Make sure to change that back if you don't want to squash and merge every time you finish a code review.

Luckily, even with the bad commits showing, the file is still only showing the simple changes that were made:



Squash and merge feature #5 — Edit

Open majorguidancesol... wants to merge 6 commits into `master` from `SquashAndMergeFeature`

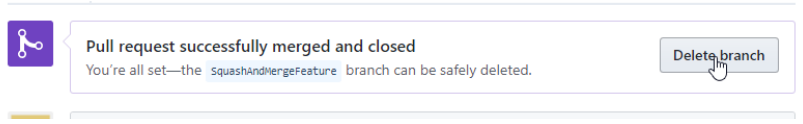
Conversation 0 | Commits 6 | Files changed 1

Changes from all commits ▾ Jump to... ▾ +3 −1 Unified Split Review changes ▾

4 info.txt View

```
@@ -11,4 +11,6 @@ and here is another change.
 Squash And Merge commit #1
 Squash and Merge commit #2
 Squash and Merge commit #3
-Squash and Merge commit #4
+Squash and Merge commit #4
+
+After Squash And Merge, and merge master.  commit #5
```

Go ahead and do a regular merge or a squash and merge if you want just one more commit. This time, delete the branch on completion:

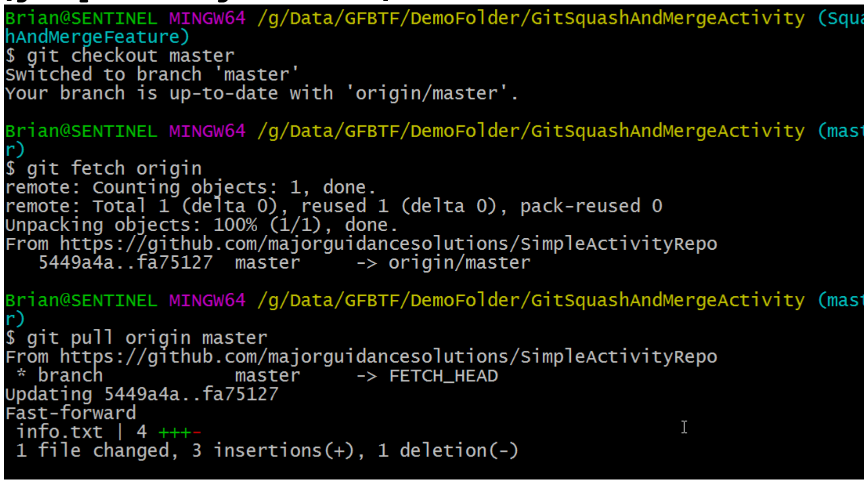


Pull request successfully merged and closed
You're all set—the `SquashAndMergeFeature` branch can be safely deleted.

Step 4: Repeat all of the operations from step 2. This time, delete the feature branch after merge.

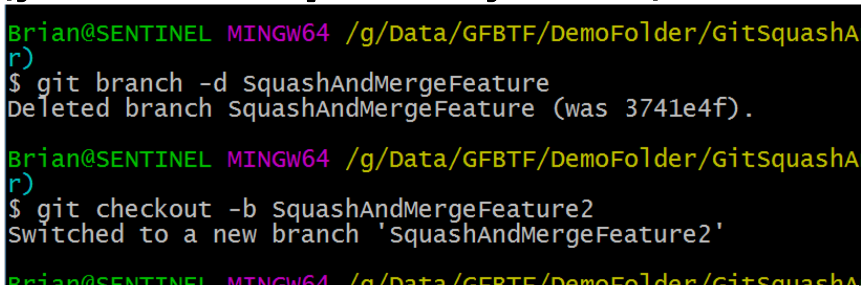
a) Get our repo up to date

   [**`git checkout master`**]
   [**`git fetch origin --prune`**]
   [**`git pull origin master`**]

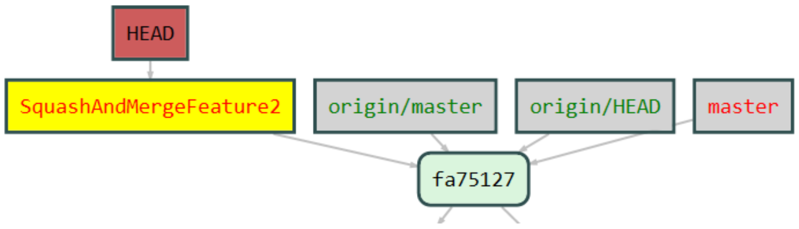


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (Squa
hAndMergeFeature)
$ git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (mast
r)
$ git fetch origin
remote: Counting objects: 1, done.
remote: Total 1 (delta 0), reused 1 (delta 0), pack-reused 0
Unpacking objects: 100% (1/1), done.
From https://github.com/majorguidancesolutions/SimpleActivityRepo
   5449a4a..fa75127  master     -> origin/master

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (mast
r)
$ git pull origin master
From https://github.com/majorguidancesolutions/SimpleActivityRepo
 * branch            master     -> FETCH_HEAD
Updating 5449a4a..fa75127
Fast-forward
 info.txt | 4 +++-
 1 file changed, 3 insertions(+), 1 deletion(-)
```

   [**`git branch -d SquashAndMergeFeature`**]
   [**`git checkout –b SquashAndMergeFeature2`**]



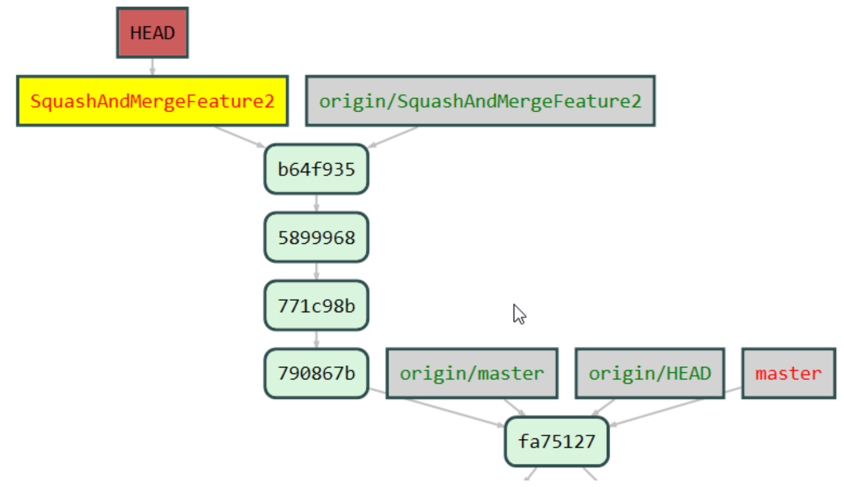
```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashA
r)
$ git branch -d SquashAndMergeFeature
Deleted branch SquashAndMergeFeature (was 3741e4f).

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashA
r)
$ git checkout -b SquashAndMergeFeature2
Switched to a new branch 'SquashAndMergeFeature2'
```

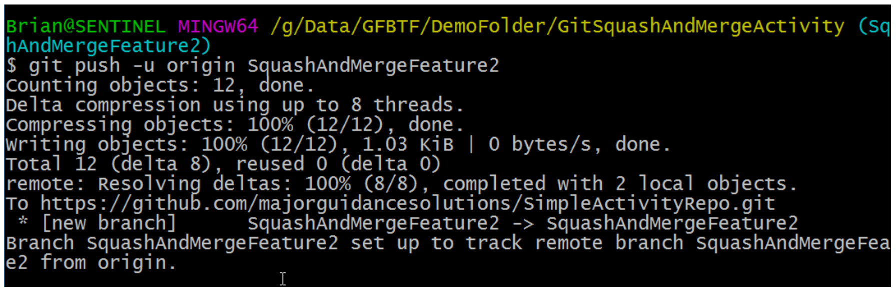


b) Perform four commits on the feature, push to GitHub, create PR, squash and merge it.

   [`code info.txt`] //leave it open
   [**`git commit –am "Squash and Merge #6`**]
   [make changes]
   [**`git commit –am "Squash and Merge #7`**]
   [make changes]
   [**`git commit –am "Squash and Merge #8`**]
   [make changes]

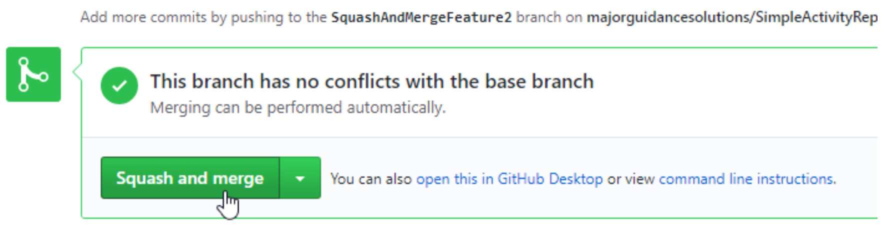
[git commit –am "Squash and Merge #9]





```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (Sq
hAndMergeFeature2)
$ git push -u origin SquashAndMergeFeature2
Counting objects: 12, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (12/12), 1.03 KiB | 0 bytes/s, done.
Total 12 (delta 8), reused 0 (delta 0)
remote: Resolving deltas: 100% (8/8), completed with 2 local objects.
To https://github.com/majorguidancesolutions/SimpleActivityRepo.git
 * [new branch]      SquashAndMergeFeature2 -> SquashAndMergeFeature2
Branch SquashAndMergeFeature2 set up to track remote branch SquashAndMergeFea
e2 from origin.
```

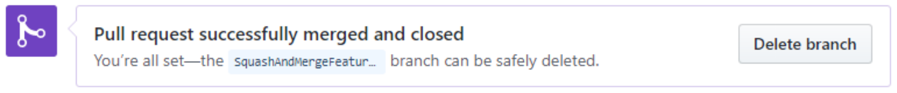
c) Get the pull request going, commit with squash and merge, delete the branch



Add more commits by pushing to the **SquashAndMergeFeature2** branch on **majorguidancesolutions/SimpleActivityRep**

This branch has no conflicts with the base branch
Merging can be performed automatically.

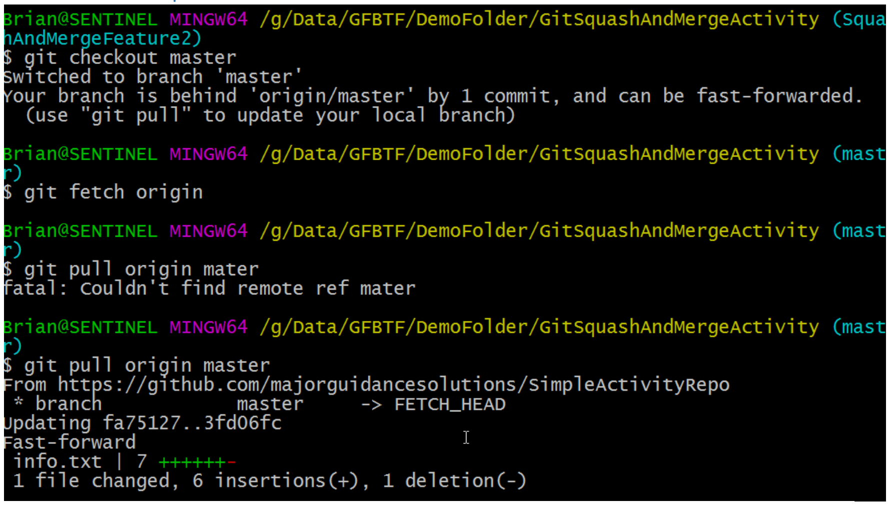
You can also open this in GitHub Desktop or view command line instructions.



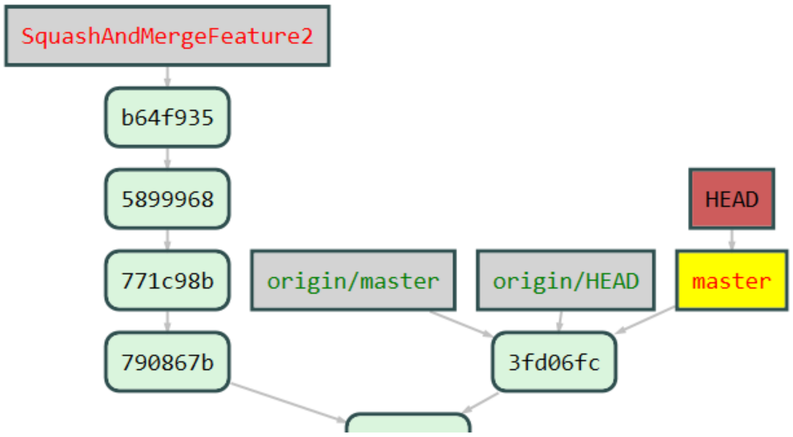
Pull request successfully merged and closed
You're all set—the SquashAndMergeFeatur… branch can be safely deleted.

Step 5: Clean up the local repo, get master up to date.

a) Get master up to date

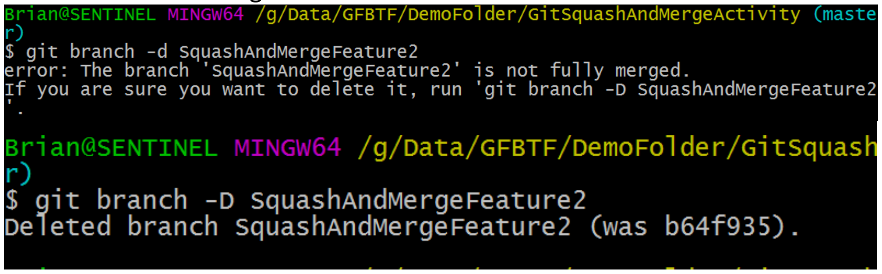


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (Squa
hAndMergeFeature2)
$ git checkout master
Switched to branch 'master'
Your branch is behind 'origin/master' by 1 commit, and can be fast-forwarded.
  (use "git pull" to update your local branch)

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (mast
r)
$ git fetch origin

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (mast
r)
$ git pull origin mater
fatal: Couldn't find remote ref mater

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (mast
r)
$ git pull origin master
From https://github.com/majorguidancesolutions/SimpleActivityRepo
 * branch            master     -> FETCH_HEAD
Updating fa75127..3fd06fc
Fast-forward
 info.txt | 7 ++++++-
 1 file changed, 6 insertions(+), 1 deletion(-)
```

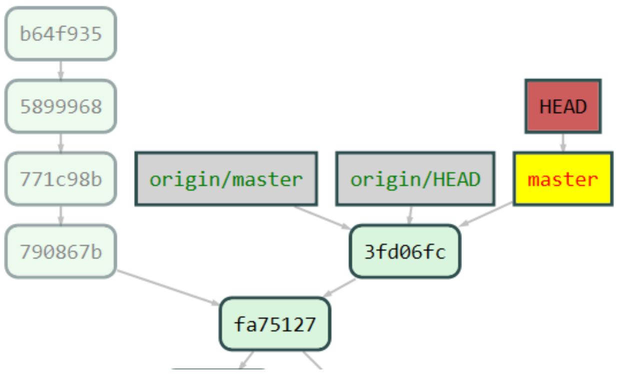


b) Delete the feature branch

We'll have to force the delete because once again we have four commits that don't line up with the history in master. If we don't use the –D option, git will warn us about unmerged commits:



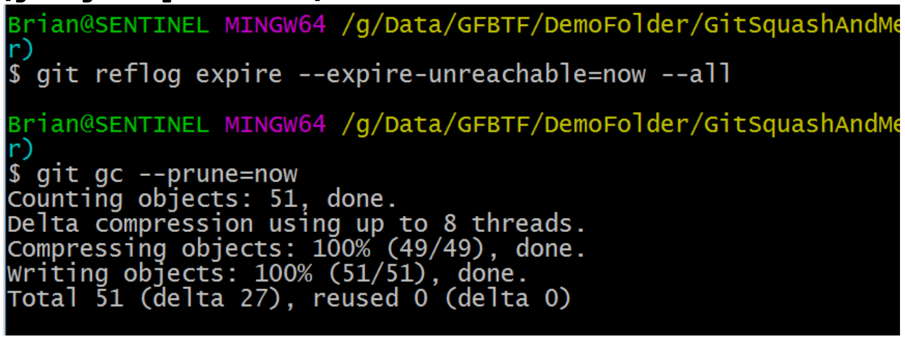
```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMergeActivity (maste
r)
$ git branch -d SquashAndMergeFeature2
error: The branch 'SquashAndMergeFeature2' is not fully merged.
If you are sure you want to delete it, run 'git branch -D SquashAndMergeFeature2
'.

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquash
r)
$ git branch -D SquashAndMergeFeature2
Deleted branch SquashAndMergeFeature2 (was b64f935).
```



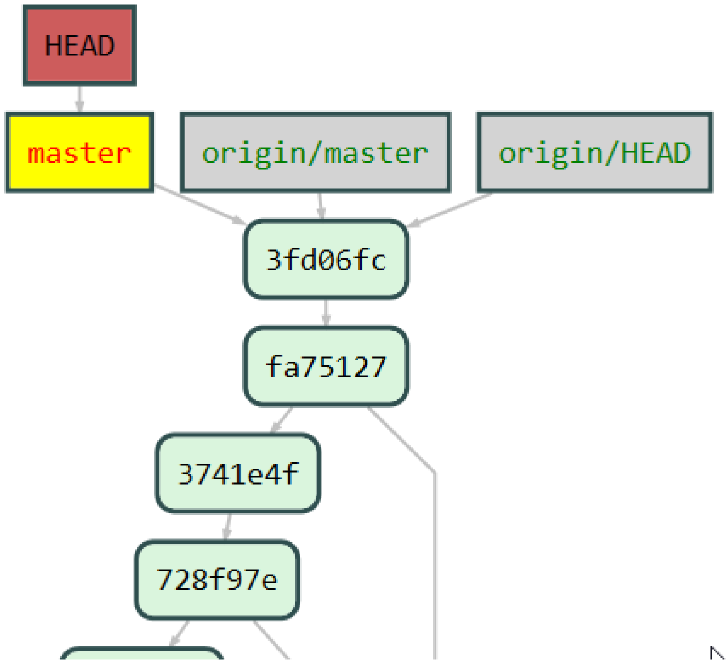
Note that we now have four unreachable commits. We can optionally clean these up.

c) Expire unreachables and garbage collect.

[**git reflog expire –expire-unreachable=now –all**]
[**git gc --prune=now**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMe
r)
$ git reflog expire --expire-unreachable=now --all

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DemoFolder/GitSquashAndMe
r)
$ git gc --prune=now
Counting objects: 51, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (49/49), done.
Writing objects: 100% (51/51), done.
Total 51 (delta 27), reused 0 (delta 0)
```



This completes our SquashAndMerge at GitHub activity.

Closing Thoughts

Squashing and merging is an easy way to get a number of commits down to just one for merge commit into the remote repository. Often, this would take place at the completion of a feature branch, when we don't care about each induvial commit, but only the entire changeset.

Since the squash and merge operation does change history, it's pretty important to delete your local branch after completing a squash and merge, simply because your repo at local won't line up with the commit history in master any longer. If you need to continue work for some reason, you could delete your branch, then open a new branch off of master which would work the same as if you had continued to develop on the branch, with no worry of having bad commit ids in the history tree.

In the end, we see that when done properly, a squash and merge is a simple way to limit the commits in your master branch and help to maintain a solid linear operational history.

This activity squashed and merged changes for multiple commits at remote during the merge of a pull request. It is also possible to squash at local before even opening a pull request, although it is a bit more involved. To squash locally, we must perform an interactive rebasing operation [covered in another activity].

Take a few minutes to make some notes about the various commands we've learned about in this activity, and practice using them.

Notes

____

____

____

____

____

____

____

____

____

____

____

____