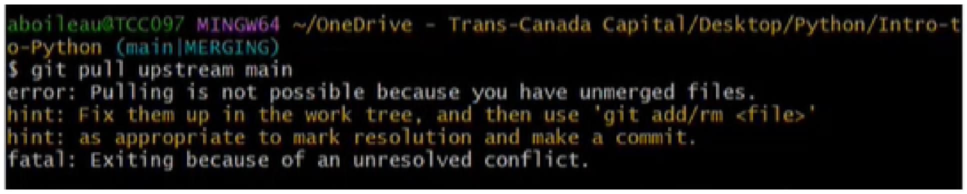
Resolving Git Merge Conflicts

Jade D.

Summer 2022

Git merge conflicts occur when the file version that exists on your computer (the local version) does not match the file version that exists online on GitHub (the remote version). You will encounter git merge issues when performing a `git pull upstream main` to pull the latest changes from the upstream (instructor's) repo.

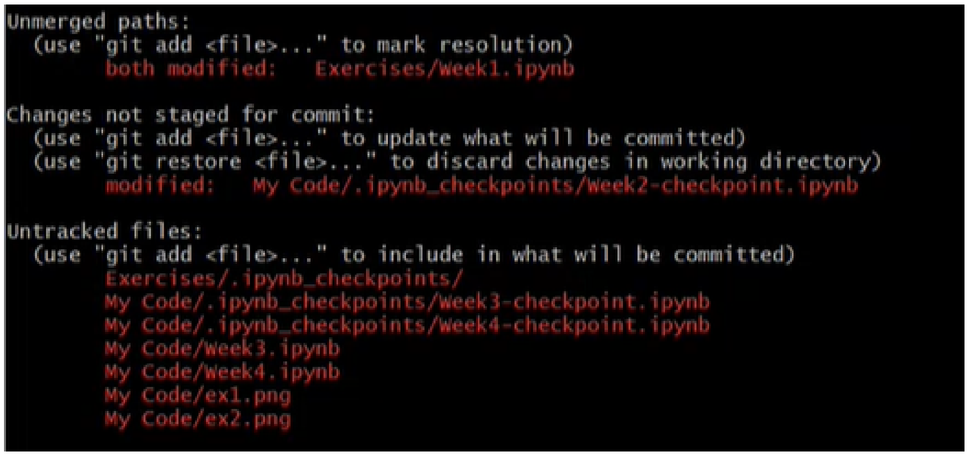
When that happens, git doesn't know which file version to keep, and which to scrap. You should receive a message like this telling you to resolve the conflict:



```
aboileau@TCC097 MINGW64 ~/OneDrive - Trans-Canada Capital/Desktop/Python/Intro-t
o-Python (main|MERGING)
$ git pull upstream main
error: Pulling is not possible because you have unmerged files.
hint: Fix them up in the work tree, and then use 'git add/rm <file>'
hint: as appropriate to mark resolution and make a commit.
fatal: Exiting because of an unresolved conflict.
```

Figure 1: Possible example of a git merge conflict message.

To figure out which file(s) is causing the merge conflict, use the `git status` command, which will produce an output like this:

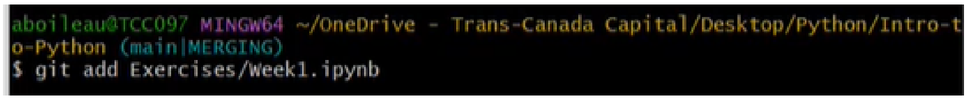


```
Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   Exercises/Week1.ipynb

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   My Code/.ipynb_checkpoints/Week2-checkpoint.ipynb

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Exercises/.ipynb_checkpoints/
        My Code/.ipynb_checkpoints/Week3-checkpoint.ipynb
        My Code/.ipynb_checkpoints/Week4-checkpoint.ipynb
        My Code/Week3.ipynb
        My Code/Week4.ipynb
        My Code/ex1.png
        My Code/ex2.png
```

Figure 2: Output of `git status` command.

Under the `Unmerged paths` section, you will see all the files that are causing the conflict. Under the `Changes not staged for commit` section, you will see all your modified files that aren't causing a merge conflict. Finally, under the `Untracked files`, you will see all your new local files that don't exist on your remote repository (these also won't cause merge conflicts).

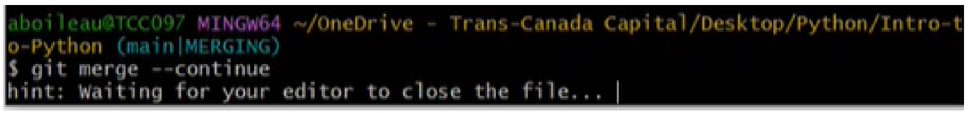
What interests us right now is the `Unmerged paths`. Following the instructions provided by git, we want to use `git add filename` to *add* the conflicting files and resolve the merge conflict. Using this command specifies that you want to keep *your* version, not the one from the instructor's repo. Use `git add filename` like this to mark the conflict resolution:



```
aboileau@TCC097 MINGW64 ~/OneDrive - Trans-Canada Capital/Desktop/Python/Intro-t
o-Python (main|MERGING)
$ git add Exercises/Week1.ipynb
```

Figure 3: Using `git add filename` to mark merge conflict resolution.

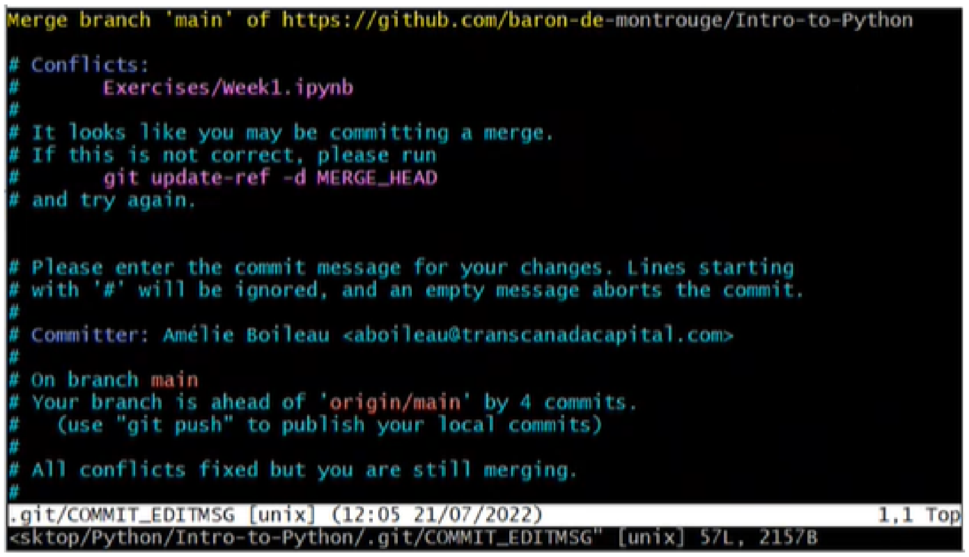
Notice the `(main|MERGING)` blue text in the console. This indicates that git is already in the process of merging the upstream repo with your local repo. If you type in `git pull upstream main` again, you will get an error message saying that you already have a merge in process. Instead, you want to tell git to continue the merge by using the `git merge --continue` command as such:



```
aboileau@TCC097 MINGW64 ~/OneDrive - Trans-Canada Capital/Desktop/Python/Intro-t
o-Python (main|MERGING)
$ git merge --continue
hint: Waiting for your editor to close the file... |
```

Figure 4: Using `git merge --continue` to continue an existing merge.

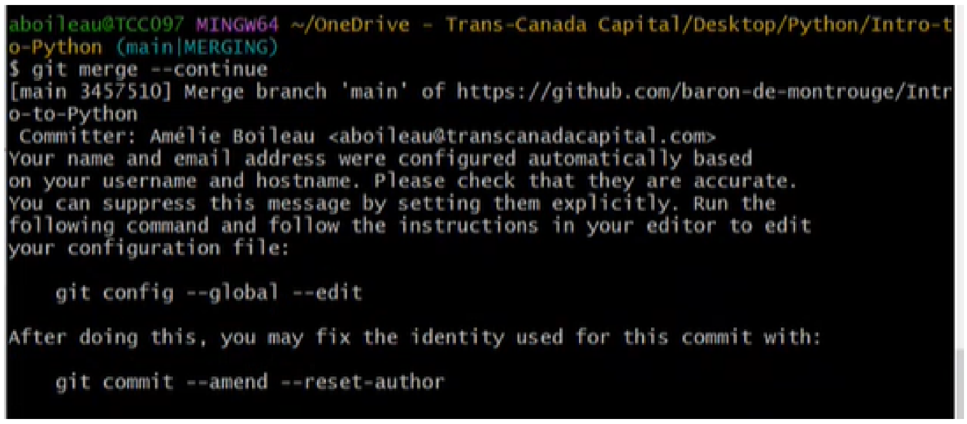
The `hint:  Waiting for your editor to close the file...` message should pop up and linger for a few seconds before a window like this appears:



```
Merge branch 'main' of https://github.com/baron-de-montrouge/Intro-to-Python

# Conflicts:
#       Exercises/Week1.ipynb
#
# It looks like you may be committing a merge.
# If this is not correct, please run
#       git update-ref -d MERGE_HEAD
# and try again.


# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Committer: Amélie Boileau <aboileau@transcanadacapital.com>
#
# On branch main
# Your branch is ahead of 'origin/main' by 4 commits.
#   (use "git push" to publish your local commits)
#
# All conflicts fixed but you are still merging.
#
.git/COMMIT_EDITMSG [unix] (12:05 21/07/2022)                          1,1 Top
<sktop/Python/Intro-to-Python/.git/COMMIT_EDITMSG" [unix] 57L, 2157B
```

Figure 5: git merge editor window

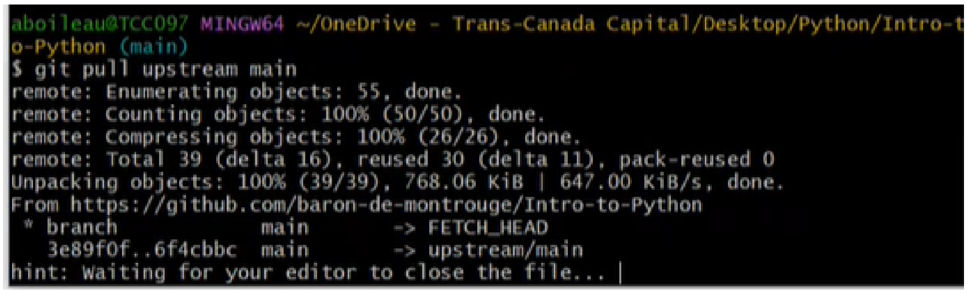
Once that window has appeared, you can type in `:wq` to close it. Finally, a window like this indicates that the merge was concluded successfully:



```
aboileau@TCC097 MINGW64 ~/OneDrive - Trans-Canada Capital/Desktop/Python/Intro-t
o-Python (main|MERGING)
$ git merge --continue
[main 3457510] Merge branch 'main' of https://github.com/baron-de-montrouge/Intr
o-to-Python
 Committer: Amélie Boileau <aboileau@transcanadacapital.com>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author
```

Figure 6: Successful git merge final window

Finally, to make sure all the most recent changes have been merged, you can go ahead and type in `git pull upstream main` one last time. You will either get a message that says that everything is `up-to date`, or you will get a message like this, indicating that there are a few more changes to merge:



```
aboileau@TCC097 MINGW64 ~/OneDrive - Trans-Canada Capital/Desktop/Python/Intro-t
o-Python (main)
$ git pull upstream main
remote: Enumerating objects: 55, done.
remote: Counting objects: 100% (50/50), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 39 (delta 16), reused 30 (delta 11), pack-reused 0
Unpacking objects: 100% (39/39), 768.06 KiB | 647.00 KiB/s, done.
From https://github.com/baron-de-montrouge/Intro-to-Python
 * branch            main       -> FETCH_HEAD
   3e89f0f..6f4cbbc  main       -> upstream/main
hint: Waiting for your editor to close the file... |
```

Figure 7: Some extra files are being merged

The editor window will appear again, and you'll need to type in `:wq` to close it and conclude the merge. You will only see a window like Figure 7 if extra files have been added to the upstream between the time the merge conflict appeared and was resolved. Usually, during your second `git pull upstream main`, you'll only get a message telling you everything is `up-to date`.

At this point, the merge conflict should be resolved, and you'll have the most recent versions of each file to work with on your local repo!