Branch Branch Branch Revolution
Simplifying your life with git branching
Outline

• Motivating Example
  • The horror story.

• What is a branch?
  • You already use them.
  • Branching visualization.
  • Git “flow.”
  • How this fixes our problem.

• How to do it.
  • Creating a new branch.
    • Using the terminal.
    • Using Github Desktop.
  • Merge/pull requests.
    • How to.

• Problem(s)!
  • Merge Conflicts
Motivating Example

- Ask Collaborator Tom to add a line in your script to run a regression.
- In the meantime you do work on some plotting.
- Try and git push and... off to stack overflow.
Horror Story

- After approximately 4 hours of searching you end up copying and pasting the code in an email, erasing the script and starting new.

- There must be a better way…

- There is, enter the git branch.
What is a “branch?”
You already use them

• Every git repo starts with a “master” branch.

• A new branch is like copying the whole repo into a new folder.

• …but a lot cleaner. (No project_newPlotVersion_1/... style shenanigans.)
Branching Visualization

- This is from a software engineering perspective but the idea holds.

https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow
Git “Flow”

- The pattern and flow of changes through the life of a project.
- Break big things into small chunks.
- For more details on how proper git flow visit this excellent writeup.
How this fixes our problem

• Collaborator Tom opens up a new “regression_branch”.

• Meanwhile you open a “plotting_branch”.

• Upon finishing Tom sends a pull request to you asking you to incorporate his changes to the base script. No more trips to S.O. to figure out pushing errors!
How To Do It:
Creating a new branch

$ git branch
* master

# Make a new branch with the current one as your template:
$ git checkout -b newBranch
Switched to a new branch 'newBranch'

# Check to see if branch was made successfully
$ git branch
  master
* newBranch

• Note: You have to push to the remote repo before any of these changes are sent to github.
Using Github Desktop

Click here to Create Branch. ($ git checkout -b...)

Click here to make a pull request. (More on this later.)
Merges/Pull Requests

• You could simply do a `git merge <branchToMerge>` in the terminal… but, like much in the terminal, it can be hard to grasp what you’re doing.

• The Pull Request is like a more polite merge. It allows conversations and an overview of the changes to be merged into the branch.

• I use pull requests for (almost) every merge.
How to

From the main repo view click here.

The Best Coffee Shop In Nashville
How to, Continued…

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: aSmallChange
- Able to merge. These branches can be automatically merged.

Made my changes
Write Preview

Here is a comment about all the awesome changes I made to the code. These help the reviewer understand the contents of the pull request. Brief Description in the title, longer in the description box.

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

Create pull request

Write a description and click here to continue.
The admin (or you) will then get a message like this.
Yay, now you’re done.
If you’re a neat freak, delete the branch.

Pull request successfully merged and closed
You’re all set—the asmallchange branch can be safely deleted.

But wait… We glossed over a small detail.

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

What if this is not the case…?
Problems!
# Make sure everything is up to date.
$ git pull

# Make a new branch to sort out problems:
$ git checkout -b HR_Department origin/problemBranch
Switched to a new branch 'HR_Department'

# Alright, I guess we have to merge in the command line...
$ git merge master

This will tell you about conflicts and git will kindly put little annotations in your files where the conflicts are.
Merge Conflicts, Continued...

# Make sure everything is up to date.
$ git pull

# Make a new branch to sort out problems:
$ git checkout -b HR_Department origin/problemBranch
Switched to a new branch 'HR_Department'

# Alright, I guess we have to merge in the command line...
$ git merge master

Now go through the files that have conflicts and find the funky <<<<<<<<<<<, =========, >>>>>>>>> lines*.

Modify the code or text so that it works how you want.

*Made that way because no language has these characters so they won't cause problems.
Merge Conflicts, Continued…

# Go back into the master branch.
$ git checkout master

# Merge with your freshly cleaned conflict branch.
$ git merge --no-ff HR_Department

# Push it up to github.
$ git push origin master

# Be happy.

For a more detailed reference on dealing with conflicts see my blog post here:

http://nickstrayer.me/nashvilleBioStats//2016/01/branching.html
¿Preguntas?