Visualizing Git Workflows

A visual guide to 539 workflows
Table of Contents

Notation

Collaboration Without Review or Branches

Merge Conflicts

Requesting Code Review

Collaboration with Multiple Branches
Notation
Notation

This is a single commit. The text is the commit ID.

This is a branch (i.e. sequence of commits). Each branch has a different color.
Notation

These boxes show the state of a repository at a given location.
This means that GitHub is storing two branches. One has 4 commits, the other has 3.
This means that Alice’s computer is storing local copies of the same two branches.
Notation

Local Copy

Remote Copy

Same branches, different commits.
Notation

I show branches like this:

They mean the same thing

Other tutorials may show this:
Collaboration Without Review or Branches
There is a “master branch” on github with two commits.
Alice clones the repository on her computer.
Bob clones the repository on his computer.
Alice commits new code to her master branch.
Alice pushes the feature branch to GitHub for all to see.
Bob fetches Alice’s new changes from GitHub.
Bob builds some of his code on Alice’s new commit.
Bob pushes his code to GitHub for Alice to see.
Alice pulls Bob’s changes, so now everyone has the same code.
Merge Conflicts

What happens when we change the same file?
Imagine Alice is fixing a bug. She starts by checking out a new purple branch, based off of her local master.
Around the same time, bob pushes a new commit, C4, to master.

Now the remote master (on GitHub) is out of sync with Alice’s local master branch.
Alice creates a new commit, C5, to fix the bug on her new branch.
Now Alice pulls Bob’s updated changes from GitHub to her computer.

Bob put the new feature on the master branch, so when Alice pulls, it goes to her local master branch.

Notice how the last commit of the bug fix branch is different from master.
Now Alice adds her local bug fix to her local master branch.

She does this by merging her changes from her bug fix branch to her master branch.

If there is a conflict with Bob’s changes, this results in a special `merge commit`. 
Requesting Code Review

Do this for every change outside of SkunkWorks!
Alice creates a new “feature” branch.
Alice commits new code to her feature branch
Alice pushes the feature branch to GitHub for all to see.

She creates a pull request and asks for a review.
After addressing a few comments in a new commit, the Koz says “Ayyyyy that code looks good.”
After Alice addresses any review comments and obtains approval, the Koz merges her changes to master.
Alice now pulls the new master from GitHub, and deletes her old branch. Now everyone has the same code.
Collaboration With Multiple branches
There is a “master branch” on github with two commits.
Both Alice and Bob clone the repository on their computers.
Alice creates a local orange branch.

Bob creates a local purple branch.
They both commit new code to each of their branches.
Alice pushes the feature branch to GitHub for all to see.

She requests a pull request.
The course staff merges Alice’s changes with the master branch on GitHub.
Both Alice and Bob pull down the changes from master.
Bob merges master into his feature branch.
Bob pushes his feature branch to master and asks for a PR.